DOCUMENT RESUME

ED 054 702	48	FL 002 627

AUTHOR Price, James D.

TITLE A Computerized Phrase-Structure Grammar of Modern Hebrew: Part I, Complex-Constituent Phrase-Structure Grammars. Final Report.

INSTITUTION Franklin Inst. Research Labs., Philadelphia, Pa.

SPONS AGENCY Office of Education (DHEW), Washington, D.C. Bureau of Research.

BUREAU NO BR-9-7722

PUB DATE Jun 71

CONTRACT OEC-0-9-097722-4411

NOTE 63p.

EDRS PRICE MF-$0.65 HC-$3.29

DESCRIPTORS Algorithms; *Computational Linguistics; *Computer Programs; Deep Structure; *Grammar; *Hebrew; Instructional Materials; Language Patterns; Language Research; Language Universals; Phrase Structure; Semitic Languages; Sentence Structure; Syntax; Tables (Data); Teacher Education; Transformation Generative Grammar; *Transformation Theory (Language)

ABSTRACT

This first part of a four-part report of research on the development of a computerized, phrase-structure grammar of modern Hebrew presents evidence to demonstrate the need for material to train teachers of Semitic languagues in the theory of grammar. It then provides a discussion of the research already done on the application of computational grammars to artificial and natural languages. Research procedures are discussed. Following a section on computational grammars, there is discussion of grammar theories and of several grammars which might be suitable for generating and analyzing Hebrew sentences. The general requirements of complex-constituent-phrase structure grammar are outlined and methods for applying it to Semitic languages are discussed. A list of references is provided. For related reports see FL 002 628, FL 002629, and FL 002 630. (VM)

ED054702

OE/115

Title VI, NDEA, Section 602 PA-48 Bureau#9-7722

FL

FINAL REPORT

Project No. 097722

Contract No. OEC-0-9-097722-4411

Franklin Institute Report No. F-C2585-1

A COMPUTERIZED PHRASE-STRUCTURE GRAMMAR OF MODERN HEBREW

PART I

James D. Price
The Franklin Institute Research Laboratories
20th and Race Streets
Philadelphia, Penn. 19103

June 1971

U. S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE

Office of Education
Institute of International Studies

FL002627

FINAL REPORT

Project No. 097722

Contract No. OEC-0-9-097722-4411

Franklin Institute Report No. F-C2585-1

A COMPUTERIZED PHRASE-STRUCTURE GRAMMAR OF MODERN HEBREW

PART I

Complex-Constituent Phrase-Structure Grammars

James D. Price
The Franklin Institute Research Laboratories
Philadelphia, Penn. 19103

June 1971

The research reported herein was performed pursuant to a contract with the Office of Education, U.S. Department of Health, Education and Welfare.

U. S. DEPARTMENT OF
HEALTH, EDUCATION, AND WELFARE

Office of Education
Institute of International Studies

SUMMARY — VOLUMES I-IV

Over the past several years, The Franklin Institute Research Laboratories has conducted research on the application of computational grammars to natural and artificial languages. Research in natural languages has been confined to the Semitic branch, modern Hebrew in particular. This report describes the results of the most recent research to help meet the need for material to train teachers of Semitic languages (especially Hebrew) in the theory of grammar and to provide basic computerized tools for further linguistic research in Semitic languages. The material developed provides the foundation, framework, and some of the basic building blocks, but many additions, corrections, and improvements must yet be made. The basic computerized research tools provided, however, will greatly facilitate the ultimate completi of the material.

This report of the development of a Computerized Phrase-Structure Grammar of Modern Hebrew has been prepared in four parts. Part I presents evidence to demonstrate the need for material to train teachers of Semitic languages in the theory of grammar. Transformational theory is shown to be the best for this purpose. The background of the present project is given together with a survey of related research and a description of the procedures involved in carrying out the research. A discussion of the theory of grammar follows in which various other types of structural grammars are examined. It is concluded that each type uses a different property of sentences as a basis for describing a language; that the other properties become restrictions on the selected property; that, granted sufficient restrictions, each type can describe a language equally well; and that several of the most prominent grammars may be viewed as highly restricted phrase-structure grammars which may be considered "transformational" grammars.

This conclusion is verified by adding restraints to a simple phrase-structure grammar sufficient for it to describe Semitic languages. The resultant grammar is called a complex-constituent phrase-structure grammar because of the set of subscripts added to the symbols. This grammar has the power to explain the common deep-structure relationships that exist between such forms as the *active* and *passive* voices by showing that they originate from different options of the same symbol. With a few simple rules in phrase-structure notation, it has the power to explain the universal patterns of a language that transcend the bounds of phrases. By the use of semantic subscripts, it has a type of *context sensitivity* sufficient for explaining the semantic concord found in natural languages. All of this is provided by a relatively small number of unordered rules without a second system of notation (i.e., without one system for phrase-structure rules, and another for transformational rules). Finally, the

general requirements of this grammar are outlined, and methods for applying it to Semitic languages are discussed.

Part II describes in detail the application of this generalized complex-constituent phrase-structure grammar to modern Hebrew. It was found to be suitable for accurately defining the syntax and orthography of a Semitic language and for mechanization on a computer. This was demonstrated by the high degree of success achieved in producing a computerized algorithm for generating Hebrew sentences (Part III), in producing a computerized algorithm for analyzing Hebrew sentences (Part IV), and in testing the rules of the Hebrew grammar by means of the computer. Of the 47 sentences generated, 42 were grammatically correct, two were correct except for a superfluous period, and three contained errors that require future modification of the rules. In the process of generating these sentences, a large percentage of the rules were tested, and in numerous cases the rules were modified to correct deficiencies and errors in their original version.

Part III describes in detail a computerized algorithm for generating Hebrew sentences, and Part IV presents a computerized algorithm for analyzing Hebrew sentences. Parts III and IV include flow diagrams, a listing of the computer programs in FORTRAN IV, and instructions for their use. The algorithms were used to test and demonstrate the Hebrew grammar, the results of which indicate that the grammar of Hebrew is essentially correct, but that some of the rules are in need of further development. In all cases where errors occurred, they were due to the content of the rules and not to the form of the grammar. Although further development is needed in some areas of the grammar, the results of the research provide good reason to believe that the generalized grammar can be successfully applied to other Semitic languages such as Arabic.

ABSTRACT

This is Part I of a four-part report of research for the development of a Computerized Phrase-Structure Grammar of Modern Hebrew. This part of the report presents evidence to demonstrate the need for material to train teachers of Semitic languages in the theory of grammar. The background of the present project is given together with a survey of related research and a description of the procedures involved in carrying out the research. A discussion of the theory of grammar follows in which it is shown that several of the existing computational grammars of natural languages may be viewed as highly restricted phrase-structure grammars and thus as of approximately equal merit. Finally, the general requirements of one of these grammars, a complex-constituent phrase-structure grammar, are outlined, and methods for applying it to Semitic languages are discussed. In subsequent parts, the generalized grammar is applied to modern Hebrew and demonstrated by computer tests to be suitable for accurately defining the syntax and orthography of a Semitic language and for implementation on a computer.

ACKNOWLEDGMENTS

The author would like to acknowledge the valuable assistance of Professor Esra Shereshevsky, Chairman, Department of Hebrew and Near Eastern Languages and Literatures, Temple University, who served as consultant. Prof. Shereshevsky carefully reviewed the manuscript of Part II, made many helpful suggestions regarding the grammar of Hebrew syntax, and checked countless details of the grammar of Hebrew orthography.

The author is also grateful to the members of the faculty of Dropsie University, especially Dr. Federico Corriente, Professor of Semitic Linguistics, who supervised the work performed under subcontract with that institution, and Mr. Ezra Cohen who classified the 1040 most commonly used Hebrew words contained in the computerized Hebrew-English Dictionary (Appendix A, Part II) and made numerous helpful suggestions regarding the Hebrew grammar.

TABLE OF CONTENTS

Page

SUMMARY — VOLUMES I-IV iii

ABSTRACT v

ACKNOWLEDGMENTS vi

PART I. COMPLEX-CONSTITUENT PHRASE-STRUCTURE GRAMMARS 1-1

1. BACKGROUND 1-1

1.1 Need 1-1

1.1.1 Need for a Theory of Grammar 1-1
1.1.2 Transformational Theory 1-1

1.2 Previous Research 1-7

1.2.1 Research at The Franklin Institute Research Laboratories 1-7
1.2.2 Other Related Research 1-11

2. RESEARCH PROCEDURES 1-12

2.1 Objective 1: Develop Algorithm for Generating Hebrew Sentences 1-12
2.2 Objective 2: Develop Algorithm for Analyzing Hebrew Sentences 1-14

3. THEORY OF COMPUTATIONAL GRAMMAR 1-15

3.1 Language, Information, Structure and Grammar . . . 1-15
3.2 Structural Grammars 1-18

3.2.1 Dependency Grammars 1-18
3.2.2 Categorical Grammars 1-19
3.2.3 Phrase-Structure Grammars 1-21
3.2.4 Predictive Syntactic Analysis 1-24
3.2.5 Transformational Grammar 1-24
3.2.6 Phrase-Structure Grammar with Complex Constituents 1-27

TABLE OF CONTENTS (Con't)

Page

3.2.7 String Transformational Grammars 1-28

3.3 Comparison of Grammars 1-29

4. COMPLEX-CONSTITUENT PHRASE-STRUCTURE GRAMMARS 1-29

4.1 Description 1-30

4.1.1 Simple Phrase-Structure Grammars 1-30
4.1.2 Illustration of a Simple Phrase-Structure Grammar 1-31
4.1.3 Limitations of Simple Phrase-Structure Grammars 1-33
4.1.4 Complex Constituents Overcome Limitations . 1-43

4.2 General Requirements 1-43

4.2.1 Symbols. 1-44
4.2.2 Subscripts 1-45
4.2.3 Rules 1-47
4.2.4 Operational Functions 1-48
4.2.5 Input Function 1-49

5. CONCLUSION 1-50

REFERENCES 1-51

LIST OF FIGURES

Fig. No.		Page
1-1	Sample Output of Program SENSYN	1-9
1-2	Sample Output of Program ANALYZ	1-10
1-3	Illustration of Deep Structure	1-17
1-4	Dependency Grammar	1-20
1-5	Tree Diagram of Phrase-Structure Grammar	1-23

LIST OF TABLES

Table No.		Page
1-1	Inflection of English Personal Pronoun	1-40
1-2	Inflection of English Present Tense Verb *Walk*	1-40

PART I

COMPLEX-CONSTITUENT PHRASE-STRUCTURE GRAMMARS

1. BACKGROUND

This part of the report presents evidence to demonstrate the need for material to train teachers of Semitic languages in the theory of grammar. The background of the present project is given together with a survey of related research and a description of the procedures involved in carrying out the research. A discussion of the theory of grammar follows in which it is shown that most of the existing computational grammars of natural languages may be viewed as highly restricted phrase-structure grammars and thus as of approximately equal merit. Finally, the general requirements for one of these grammars, a complex-constituent phrase-structure grammar, are outlined, and methods for applying it to Semitic languages are discussed.

1.1 Need

1.1.1. Need for a Theory of Grammar

In a recent paper presented at the Regional Seminar of the SEAMEC Regional English Language Centre in Singapore, D. M. Topping[1] said it is not sufficient that a language teacher merely speak the language he teaches, rather he needs a theory of grammar and he needs to know his language from that point of view.* This does not refer to teachers of grammar, but to teachers of language. Teachers of mathematics are required to know more than the multiplication tables, and teachers of chemistry must know more than the periodic tables. The same should hold for language teachers. The next section demonstrates that transformational theory is the best for such training. In a later section, it will be shown that complex-constituent phrase-structure grammars can be considered "transformational-type" grammars, that they are well-suited for describing Semitic languages and for implementation on computers.

1.1.2 Transformation Theory

Transformational grammar was first introduced by Zellig Harris[2] and further developed to its present form by Noam Chomsky.[3,4] This theory views language as having a small set of deep structures that are defined

*References are listed at the end of this volume.

by phrase-structure rules which generate "kernel sentences" that convey information or meaning. In addition, it views language as having a small set of transformations that operate in sequence on the "kernel sentences" to produce the surface structure sentences of the language. Transformations produce perturbations of surface structure without altering meaning.

Other types of grammars treat the relationship of deep structure and surface structure from different points of view but end up with the equivalent of transformations. These include the String Analysis Grammars of Harris,[5] Joshi,[6] and Sager;[7,8] the Predictive Syntactic Analysis Grammars of Rhodes,[9,10] Kuno and Oettinger,[11,12] and Lindsay;[13] and the Complex-Constituent Phrase-Structure Grammars of Harmon[14] and Price.[15,16,17] As explained later, all these are considered transformational grammars for the present purpose.

In evaluating the implications of transformational grammar for language teaching, Topping concluded (1) that transformational grammar tells the most about a language, (2) that it is based on a good model of the human language mechanism, (3) that it is based on a good psychological theory of language learning, and (4) that it is a good guide to language education. Of course he pointed out some implications that were not relevant to the needs of language teaching, but he concluded that language teachers should know the language they teach from the transformational point of view. The following material provides supporting evidence for this conclusion.

A Comprehensive Theory of Language. Transformational theory emphasizes the distinction between deep structure and surface structure of language and defines the relationship between them, whereas non-transformational theories describe surface structure only. Transformational theory treats language as an integrated whole, whereas other theories treat phonology, morphology, and syntax as separate features. Transformational theory defines language universals, whereas others emphasize diversities. Finally, transformational theory includes semantic components in grammatical descriptions, whereas others relegate semantics to the dictionary. In all these features, transformational theory tells more about language than other theories.

A Good Model of the Human Language Mechanism. Transformational theory views the human language mechanism as a system which can be described by reference to a small set of unchanging rules and a small set of processes or transformations. Conversely, non-transformational theories view language as a very large set of unrelated rules.

Transformational theory is able to explain how *different surface structures* convey the same meaning by showing that they are derived from the same deep structure. It is able to explain how ambiguous sentences--those with the *same surface structure*--convey different meaning by showing that they are derived from different deep structures. It is able to explain sentences with *apparently similar surface structure* by showing

different deep-structure derivations. It is also able to explain recursion, or the principle of structure within structure, on the basis of the deep-structure rules. In all these features, transformational theory explains the operations of the human language mechanism in terms of a few simple structures and processes, whereas non-transformational theories explain them in a very cumbersome way, or not at all.

A Good Psychological Theory of Language Learning. The transformational theory of language learning as discussed in works by Lenneberg,[18] Chomsky[19] and Topping[1] may be summarized as follows: (1) Human beings do not learn their native language solely through imitating and memorizing surface structures they hear. The number of surface structures an infant is exposed to during his language-forming years is enormous and varied to a degree beyond estimation. (2) Language capability is developed through the internalization of a few deep-structure rules of the language and a slightly larger number of rearranging processes, or transformation rules, which provide for converting deep structures into surface structures. (3) Language is not a set of habits, but is the result of deliberate application of cognitive processes to a finite set of rules that have been learned. This theory stands in sharp contrast with older theories that view language as "a set of habits."

Topping[1] has said that every physically sound human being is born with the capacity for producing language at certain stages of his development. He will produce sentences of a predictable structure at each stage--sentences very much like those produced by his peers. The language he produces is not an exact imitation of what he has heard, but is a product of the set of words and rules that he has induced by using his own innate language-producing mechanism. The language learning process is stimulated best when an individual is exposed to sentences that have been derived from deep structures and transformations that are in phase with his given stage of language development. Transformational theory explains these observations better than other theories.

Adults who study their native language will best understand it if they are taught to recognize the elements, rules, and processes that make up their innate language mechanism. This is not necessarily accomplished by formal procedures such as the axioms and theorems of mathematics, but by presenting the structures of language in such a way to produce a conscious awareness of the elements, rules and processes that constitute the mechanism. Transformational theory best explains this process.

For human beings learning a second language, the learning process is different. These students already have internalized the deep-structure rules of their native language and the transformation rules for producing sentences. They have an innately developed linguistic model of their language which they use unconsciously every day. By using their present linguistic model as a guide, they can easily associate the deep structures and transformations of the new language with those of

their native language. Although such students may not necessarily study the language in terms of sophisticated grammar, the teaching material should be prepared with a good grammatical model as a guide--one that matches the innate human model.

A Good Guide to Language Education. Language education material that presents the students with opportunities to make use of the the innate cognitive processes by which they organize their own native language system will be a much greater stimulus to the learner than material which requires them to repeat and memorize. Transformational grammars are based on the best model of the human language mechanism and on the best psychological theory of language learning, thus they can be used as a guide for producing language education material. The phrase-structure rules of the deep structure define the simplest constituents of the language. The transformations (or equivalent) provide a key to classifying degrees of complexity. Those constituents requiring the least number of transformations are the least complex. The language universals and the semantic components can be used to call attention to similarities between the second language and the native language. None of these features is easily available in non-transformational grammars.

Transformational grammars enable educators to arrange language texts for children in phase with their language development and thus to expose the children to sentences that have been derived from deep structures and transformations that best stimulate the language learning process at their given stage of development. They enable educators to arrange language texts for adults who study their native languages so that they recognize the elements, rules, and processes that make up their innate language mechanism. They enable educators to arrange language texts for adults learning a second language so that they may easily associate the deep structures and transformations of the new language with those of their native language. In all these features transformational grammars are better than non-transformational grammars.

Objections to Transformational Grammars. Not all language educators are equally convinced of the merits of transformational grammars. Their objections and reservations may be summarized by the statement of Carleton T. Hodge, Professor of Linguistics and Anthropology at Indiana University: "There is, in the first place, no generally accepted linguistic model for [grammar]. The transformational generative approach is in more constant flux than prior models. It has, however, produced some useful grammars of uncommon languages, though the format is too forbidding for the general reader and most other students of the language. This is true of some other approaches also, and the problem of informative presentation is yet to be solved."[20]

Although the first objection--that there is no generally accepted linguistic model of grammar--is true, the fact remains that all the leading models are based on some variety of transformational theory. The major difference between the models is one of notation and not of

theoretical basis. Each is able to produce the equivalent of the other by appropriate manipulation of symbols. It is more important that work on a language proceed along one of these lines rather than wait until one notation variant becomes dominant.

The second objection--that the transformational generative approach is in more constant flux than prior models--is true because the theory is relatively new and still in the developing stage. However, the areas of flux are those that define the finer details of the theory. The basic principles that will best benefit the training of language teachers are well established. Future research will crystallize the finer details, but educators should not postpone the use of the established principles until such time.

The third objection--that the format of transformational grammars is too forbidding for the general reader--is also true of other approaches as Hodge admits. This same objection could be made of other formalized disciplines such as mathematics, logic and chemistry. However, these disciplines are still taught to advanced students, particularly those who become teachers. The same should be true for language teachers. They should not be deprived of the advantages provided by studying the language they teach from the transformational point of view.

It is important to note, however, Hodge's statement that the transformational generative approach has produced some useful grammars of uncommon languages.

1.1.2.1 Transformational Material For Commonly Taught Languages

Material is available for training teachers of the commonly taught languages from the transformational point of view. The following research projects are listed by the Center for Applied Linguistics[21] as applying transformational grammar to the indicated languages:

English:	Robert P. Stockwell, UCLA Judith Anne Johnson, Univ. of Michigan
French:	Antonio A. M. Querido, Univ. of Montreal
German:	Henri Wittman, McGill Univ., Montreal
Hungarian:	Sándor Károly, Hungarian Acad. of Science, Budapest Fereno Kiefer, Hungarian Acad. of Science, Budapest

In addition, the Center lists the following research projects in transformational theory, most of which are applied to English.

P. Stanley Peters Jr., Univ. of Texas
Elizabeth F. Shipley, Eastern Pa. Psychiatric Institute, Philadelphia
Susumu Kuno, Harvard Univ.
Joyce Friedman, Univ. of Michigan

Many other research projects that are not listed under the descriptor "transformational theory" are applying transformational-type grammars to such languages as Russian, German, French, and English. These include the previously cited research of Chomsky, Harris, Joshi, Sager, Rhodes, Kuno and Oettinger, Lindsay, Harmon, and others.

Some researchers are applying transformational grammar directly to the teaching of languages, for example, Wittman[21] with German. Many others are making use of transformational grammar indirectly in the teaching of languages.

It is evident that much material is available and being used for training teachers of the commonly taught languages from the transformational point of view. The next section demonstrates the need for such material for the less commonly taught languages such as Arabic and Hebrew.

1.1.2.2 Need for Transformational Material for Uncommonly Taught Languages

The original assessment made by the Office of Education, under the National Defense Education Act, rated Arabic as one of the five critical uncommonly taught languages for the United States.[22] These five languages together with Hebrew accounted for 25,051 registrations in 1968.[23] Of these registrations, 45 percent were in Semitic languages (Arabic and Hebrew).

Although in the original assessment made by the Office of Education Hebrew was not listed as one of the five uncommonly taught languages that is critical for the United States, it has become increasingly important in the last few years. Kant[23] listed 10,169 registrations for Hebrew in 1968, the largest number of any of the less widely taught languages. This was an increase of 265.2 percent over the number of registrations in 1960, and it seems certain that this rapid growth in registrations will continue for some time. Gage[22] lists modern Hebrew along with Mandarin, Japanese, and Portuguese as the four most important of the neglected languages, with Norwegian, Swedish and Arabic forming the second most important group. He further states, "It seems dubious, however, that the study of the critical languages is as yet broadly based enough to make up the U. S. deficit of people able to operate in them relative to anticipated needs."[22] Under these circumstances it is clear that there is a need for training more teachers, and their training should include material from the transformational

point of view. The material provided in this report is a partial fulfillment of this need.

1.2 Previous Research

1.2.1 Research at The Franklin Institute Research Laboratories

For several years research has been conducted at The Franklin Institute Research Laboratories on the application of computational grammars to natural and artificial languages. The first phase of the work involved the development of a generalized, complex-constituent, phrase-structure grammar as a tool for linguistic research. The grammar appeared to be very powerful for use in the study and teaching of natural-language grammar and syntax.

The second phase of the work involved testing and demonstrating the power of the grammar to generate the correct orthography of inflected words of a natural language. To do this, a complex-constituent, phrase-structure grammar was written for the orthography of modern Hebrew words.[16] The work consisted of a complete analysis of Hebrew morphology using modern Hebrew orthography (i.e., no vowel points). The grammar turned out to be very simple, consisting of seven rules, seven look-up tables, and a dictionary. It uses one initial symbol and six terminal symbols (no intermediate symbols) with 11 complex descriptors, and is capable of producing the correct orthography of any Hebrew word from a complete grammatical description of the word. The grammar was reduced to algorithm form, and its operation was programmed on a computer. It was then tested on a computer and found to produce the correct orthographies of all words tested, with no errors and no ambiguities.

The third phase of the work involved testing and demonstrating the power of the grammar to analyze the inflected words of a natural language. To do this, the rules of the Hebrew word-generating grammar were written in reverse. A few additional rules, symbols, and descriptors were required to account for compound words. Again the grammar was relatively simple, consisting of ten rules, 15 look-up tables, and a dictionary. It uses one initial symbol and nine terminal symbols with up to 14 complex descriptors. This grammar was reduced to algorithm form and tested.[17] The algorithm is capable of computing one or more complete grammatical descriptions for any Hebrew word. The description includes root, stem, number, gender, person, and all other grammatical attributes. Programming flow charts were made, and the algorithm was manually tested and found to be correct, with no errors or ambiguities. The economizing techniques used in the algorithm assure the pursuit of highly probable paths and the abandonment of unfruitful paths.

The fourth phase of the work consisted of testing and demonstrating the power of the generalized grammar to generate sentences in

a natural language. To do this, a transformational-type, complex-constitute, phrase-structure grammar of modern Hebrew syntax was written.[15] The grammar consisted of approximately 180 rules using one initial symbol and 20 terminal symbols with up to 17 complex descriptors. It was capable of producing an infinite variety of sentences. It did not produce all possible sentences in Hebrew, but covered most of the commonly used types of sentences.

As part of the present project, this grammar was implemented and tested on a computer and thoroughly revised and corrected to incorporate most of the research findings. The resultant grammar is contained in Part II of this report. Although research should be continued, the grammar can be used in its present form for training teachers of Hebrew.

Two computer programs that serve as valuable research tools were also developed during this project. The first program, SENSYN, is an algorithm for generating Hebrew sentences; the second program, ANALYZ, is an algorithm for analyzing Hebrew sentences. The use of the computer demands that the grammar rules be defined to a degree of precision never before required. As a result, many less obvious features of the language have been discovered, and many improvements and corrections have been made in the grammar.

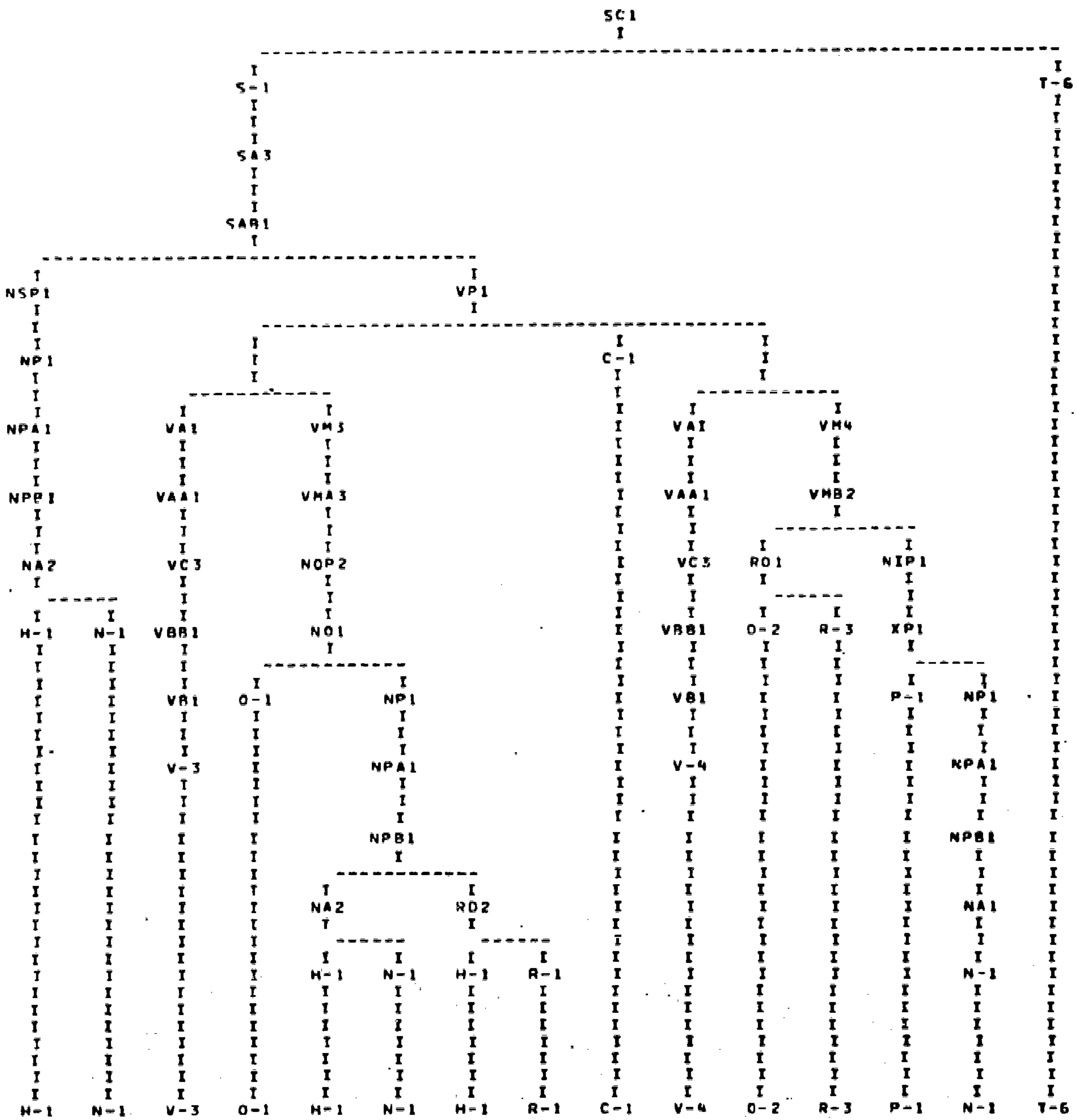
Program SENSYN, the algorithm for generating Hebrew sentences is presently being used to construct Hebrew sentences automatically. The program reads in a grammatical description of the desired sentence, and by making use of the rules of the Hebrew grammar, computes the correct syntactic order of each word of the sentence and the correct orthography (spelling) of each word in transliterated English characters. It then constructs a tree diagram of the generated sentence and writes the Hebrew sentence in transliterated characters. Figure 1-1 is a sample of the output of the program. This program is fully described in Part III of this report. Section 2.3.1 of Part II contains additional examples that demonstrate the power and versatility of the program.

Program ANALYZ, the algorithm for analyzing Hebrew sentences, is presently being used to analyze the syntax of Hebrew sentences automatically. The program reads in a grammatical description of each word of the sentence, and, by making use of the rules of the Hebrew grammar, computes a syntactic analysis of the sentence, constructs a tree diagram of the analysis, and writes out a sequence of sentences in English which are exhaustive descriptions of each constituent of the analysis. Figure 1-2 is a sample of the tree diagram output of the program. Section 2.3.2 of Part II contains additional examples together with the associated English description of the analyses. (This program is fully described in Part IV of this report.)

These two programs, as well as the Hebrew grammar, can be used for training teachers and research workers in the field of computational linguistics.

EQUIVALENT ENGLISH SENTENCE-
THE STUDENTS LEARNED THE POEMS AND KNEW THEM BY HEART.

TREE DIAGRAM OF HEBREW SENTENCE



RESULTANT HEBREW SENTENCE-

HTLMYDYM LMDW AT HSYRYM HALH WYDOW AWTM OL PH.

Figure 1-1. Sample Output of Program SENSYN

HEBREW SENTENCE ANALYZED--
HTLMYDYM LMDW AT HSYRYM HALH.

TREE DIAGRAM OF HEBREW SENTENCE No. 12.

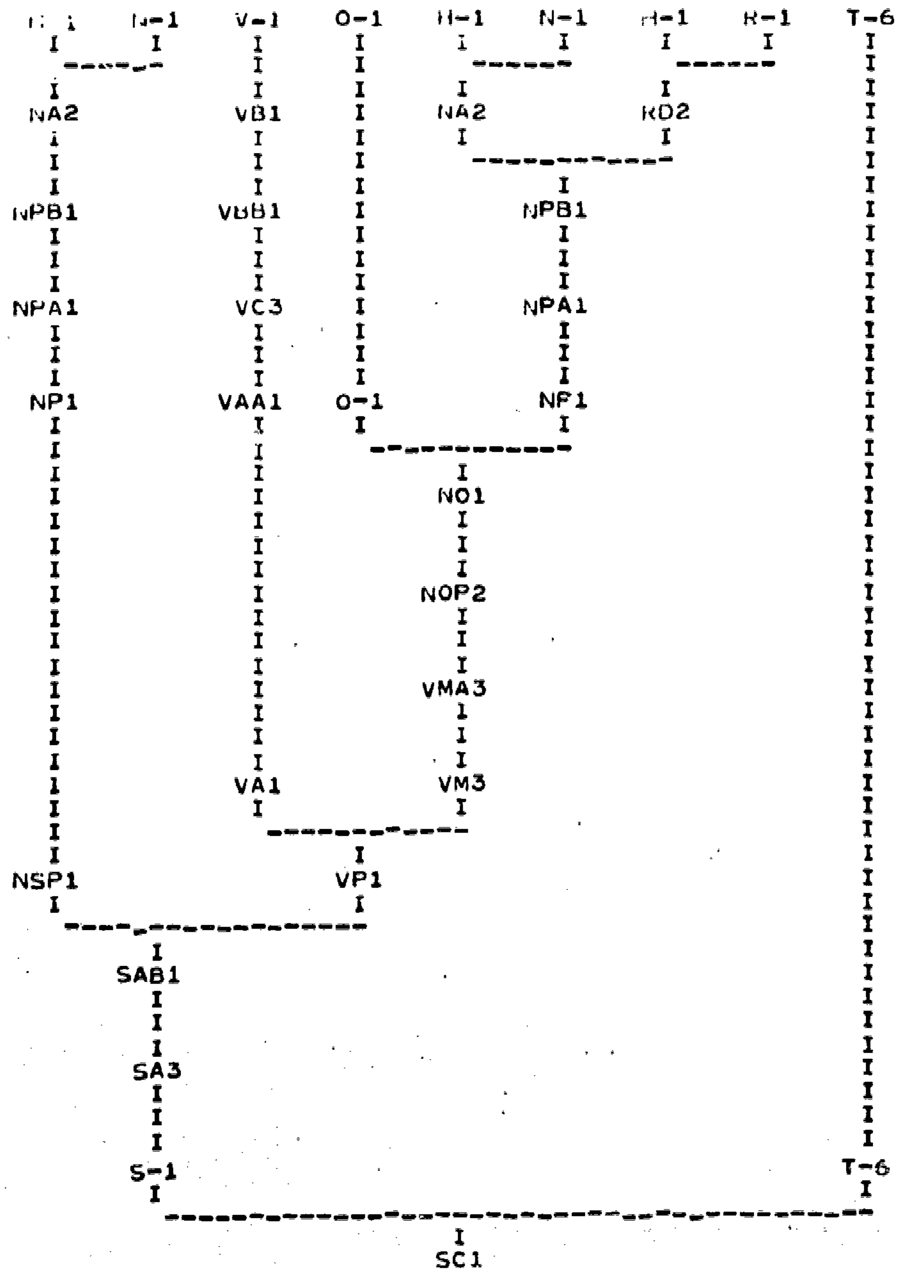


Figure 1-2. Sample Output of Program ANALYZ

1.2.2 Other Related Research

Work on the application of computational linguistics to Hebrew has been reported from the University of Texas.[24-27] This work consists of computer processing for studies conducted by Dr. Paul Samuelsdorff at the University of Cologne, Germany. A notice in the *ICRH Newsletter*[28] describes the work dealing with word order, ambiguity, and inserting the article and copula. Dr. Samuelsdorff describes the work in an article in *Forschungsberichte*.

Research conducted by Rabbi G. Lazewnik at New York University was directed at developing a stem-recognizing procedure that will enable the automatic production of a concordance of ancient Hebrew manuscripts. The work was funded by the U.S. Office of Education, Arts and Humanities Branch.

Mr. William J. Adams, Jr.[28] of the Hebrew Union College in Cincinnati is working on a computerized concordance to the Hebrew Bible in conjunction with Dr. Samuel Greengus of the Hebrew Union College and Mr. Fred Lundberg of the University of Cincinnati.

Professor Lawrence V. Berman[29] of Stanford University has utilized the Berkeley Machine Translation Project Concordance Program (TRICON) for a concordance of verbs, nouns and adjectives.

A linguistic study of the nominal phrase in Modern Hebrew which centers on the syntactic structure of nominal phrases is being undertaken by Ornan[30] at the Hebrew University of Jerusalem.

At Bar-Ilan University, Ramat-Gan, Israel, Yaacov Choueka has conducted research on the automatic grammatical analysis of Hebrew words[31-33] and on the statistical aspects of modern Hebrew prose.[34,35] In addition, Asa Kasher is conducting research on computational stylistics of Hebrew.[35]

At Indiana University, Carleton T. Hodge and his associates are preparing basic teaching materials in Chad Arabic, Tunisian Arabic and Moroccan Arabic.[35]

Arnold C. Satterthwait of Harvard University has conducted research on parallel sentence construction grammars of Arabic and English.[36,37]

At the University of Michigan, Mary M. Levy is investigating the plural of the noun in modern standard Arabic.[35]

Dr. Paul Enoch of the Technion Research and Development Foundation Ltd., Haifa, Israel, is directing a project for corpus analyses of colloquial Israeli Hebrew.[38] The objectives of the project are to establish a large corpus of words recorded from live conversations, to perform statistical analysis of the corpus and to establish word lists according to selected parameters.

Alexander Grosu of Tel-Aviv University conducted a study of the isomorphism of semantic and syntactic categories of sex and gender, number and numerosity in English and Hebrew.[39]

Ernest McCarus and associates are conducting research on the syntax of modern literary Arabic at the Center for Research on Language and Language Behavior, University of Michigan.[21]

Relativization in Hebrew from the transformational point of view has been investigated by Yehiel Hayon for his Ph.D thesis at the University of Texas.[40,41]

2. RESEARCH PROCEDURES

In achieving the following major objectives, attention was given to presenting the results of the research in a form that could be used to train teachers of modern Hebrew from the transformational point of view and to train research workers in the field of computational linguistics.

2.1 Objective 1: Develop Algorithm for Generating Hebrew Sentences

An algorithm for generating Hebrew sentences was developed which consists of a set of input variables, a set of operational functions, a set of mapping functions, and a set of output statements. This activity involved the following tasks:

1. The rules of the complex-constituent phrase-structure grammar of Hebrew syntax were completely revised and organized into an algorithm for generating Hebrew sentences. This task consisted of the following steps.

   a. The input requirements of the algorithm were determined by listing and organizing all the arbitrary decisions of the existing Hebrew grammar. The requirements consist of a general syntactic and semantic description of the sentence to be generated.

   b. The symbols of the algorithm were defined. These consist of the symbols of the Hebrew grammar which were listed and organized into computational form.

c. The operational functions of the algorithm were defined. These functions are a small set of statements that define the interrelationships of the subscripts on the symbols of the algorithm.

d. The mapping functions of the algorithm were determined. These functions are a set of approximately 180 statements that define the interrelationships of the symbols of the algorithm. They were determined by organizing the rules of the Hebrew grammar into computational form.

e. The output of the algorithm was defined. It consists of a tree diagram of the generated sentence, a listing of the generated Hebrew sentence in transliterated characters, and a listing of the equivalent English sentence (see Figure 1-1). In addition, the output contains an exhaustive grammatical description of each nodal point in the tree diagram when specified by an input option.

2. The second task of this objective was to program the algorithm to operate on a computer. This task consisted of the following steps:

   a. The main program was flow-charted and coded in FORTRAN IV programming language.

   b. Fifteen operational functions of the algorithm were flow-charted as subroutines to the main program and coded in FORTRAN IV programming language.

   c. The program was made operational on a UNIVAC 1108 computer.

3. The third task of this objective was to test the algorithm as follows:

   a. Forty-seven sentences of various types and complexities were selected for generation by the algorithm.

   b. The description of these sentences was written in terms of the input data of the algorithm.

   c. The input data of each sentence were presented to the computerized algorithm.

   d. The resultant output of each generated sentence was compared with the original sentence.

   e. All differences and all observed limitations and failures of the algorithm were noted. Any errors in the algorithm or the grammar were corrected and tests were repeated.

The resultant algorithm and tests are described in Part III and the revised grammar of Hebrew syntax is described in Part II of this report.

2.2 Objective 2: Develop Algorithm for Analyzing Hebrew Sentences

An algorithm for analyzing Hebrew sentences was developed which consists of operating the rules of the sentence-generating algorithm in reverse. It consists of a set of input variables, a set of operational functions, a set of mapping functions, and a set of output statements. The following tasks were required to accomplish this objective:

1. The rules of the complex-constituent phrase-structure grammar of Hebrew syntax were organized into an algorithm for analyzing Hebrew sentences. This task consisted of the following steps:

   a. The input requirements of the algorithm were defined. The input of this algorithm is a complete grammatical description of each word in the sentence to be analyzed.

   b. The symbols of the algorithm were defined. These symbols essentially are the symbols of the sentence-generating algorithm. However, a few new symbols were required for an analyzing procedure.

   c. The set of operational functions was determined for the algorithm. These functions define the correspondence of the subscripts of the symbols entering a computation with the subscripts of the symbols in the mapping functions. In addition, these functions define the computations to be performed on a given string of symbols.

   d. The set of mapping functions of the algorithm was determined. These functions consist of approximately 180 statements that define the interrelationships of the symbols of the algorithm. They were determined by organizing the rules of the Hebrew grammar to accommodate efficient computations in reverse.

   e. The set of output statements was defined for the algorithm. The output of the algorithm is a complete description of the syntactic analysis of the input sentence. The output also consists of a tree diagram of the resultant analysis (see Figure 1-2). In addition, the output contains an exhaustive grammatical description of each nodal point in the tree diagram when specified by an input option.

2. The second task of this objective was to program the sentence-analyzing algorithm for use on a computer. This was accomplished in the following steps:

   a. The mapping functions were flow-charted as the main program and were coded in FORTRAN IV programming language.

   b. Eleven operational functions of the algorithm were flow-charted as subroutines of the main program and coded in FORTRAN IV programming language.

c. The program was made operational on a UNIVAC 1108 computer.

3. The third task of this objective was to test the algorithm on the computer as follows:

   a. The descriptions of 26 sentences previously selected were written in terms of the input requirements of the algorithm, i.e., an exact grammatical description of each word of the sentence.

   b. The input data of each sentence were presented to the computerized algorithm.

   c. The resulting parsings were compared with those obtained by classical grammatical methods.

   d. All differences and observed limitations and failures of the algorithm were noted.

The resultant algorithm and tests are described in Part IV of this report. Consideration was given to methods for applying the generalized grammar to other Semitic languages. These methods are included in Section 1.4.2.

3. THEORY OF COMPUTATIONAL GRAMMAR

This section provides the theoretical basis for computational grammars. The general concepts of language, information, structure and grammar are considered, followed by a review of the most prominent approaches to computational grammars and a comparison of their merits.

3.1 Language, Information, Structure and Grammar

When one thinks of language and grammar, attention is directed to natural languages, such as English, in their written or spoken forms, by means of which humans are able to communicate through sequences of sounds or symbols. Grammars of these languages are recognized as sets of rules that govern the production of sequences of sounds or symbols that convey information.

In addition to natural languages, artificial languages have been invented for communicating intelligence for special information systems. For example, the set of statements in some formalized system of mathematics may be considered a language. The grammar of such a language is the set of rules that governs the production of valid statements in that system.

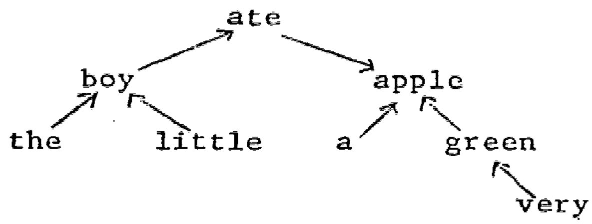
Languages, therefore, are means of communicating information in one form or another. The originator of a communication must encode the information into a sequence of symbols of a language; the recipient of the communication must decode the information from the symbols. The information itself consists of a number of discrete information (semantic) units that are interrelated in some organized fashion which is referred to herein as *deep structure.*

Figure 1-3 illustrates deep structure and shows three methods of mapping the structure of the sentence *the little boy ate a very green apple.* Method (c) is the best of the three methods because it identifies not only the various kinds of relationships that exist between the words but also the successively deeper levels of relationships between groups of words. Deep structure is part of the information and must be included in the communication.

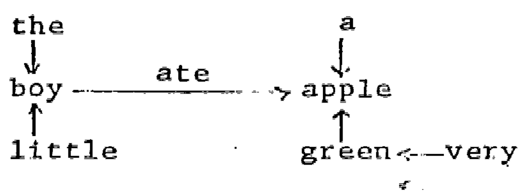
The originator of the communication must encode the information to correctly identify the information units and all structural relationships. Since languages are inherently one-dimensional (being confined to sequences of symbols) and since the information is usually multidimensional (because of the structural relationships), the language must provide symbols for both the information units and the structural units, or it must use sequential position to encode deep structure, or some combination of both. The first alternative produces long, highly inflected sentences. The second alternative requires a set of encoding rules that transform structural relationships into sequential relationships and vice versa. This is where grammars of syntax come into play. Generative syntax grammars are sets of *encoding* rules that transform deep-structure relationships into sequential relationships (surface structure); analytical syntax grammars are sets of *decoding* rules that transform sequential relationships (surface structure) into deep-structure relationships.

Natural languages use the third alternative, a combination of sequential and symbolic encoding that employs such devices as inflectional affixes, prepositions, particles, punctuation, and so forth. Highly inflected languages are less dependent on sequential encoding, providing instead redundant information that is common to structurally related words (thus the phenomenon of concord). This permits sequence to vary for the sake of emphasis or style. Because of the mixture of encoding techniques found in natural languages, structural grammars that deal only with syntax (sequential encoding) are inadequate and must be modified to account for the other encoding methods used.

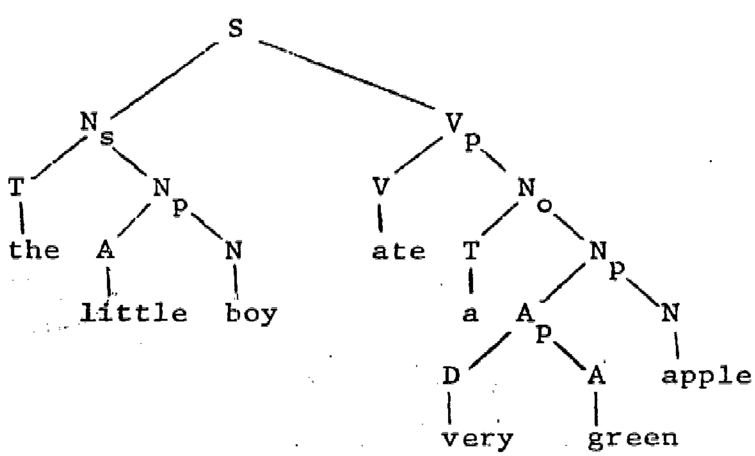
In considering grammars, it should not be surprising to find that grammars themselves can be expressed in some formalized system of notation. In fact, most artificial languages now being invented originate with some formalized grammar. In the following section, various approaches to providing formalized grammars for natural languages are summarized.



(a)



(b)



(c)

Figure 1-3. Illustration of Deep Structure

3.2 Structural Grammars

Structural linguistics deals with form or the arrangement of elements of natural languages. The structural linguist is interested in formalizing principles and methods for (1) discovering and isolating basic elements of languages and (2) writing rules for combining these elements into meaningful arrangements. Structural grammar is concerned with the latter of these interests. While it is recognized that classical grammar is basic to many European and Asiatic languages and to theories of natural languages, the structural grammarian may choose to deviate from the classical approach. He soon realizes, however, that although he speaks the same language as that spoken by the classical grammarian, there is a semantic difference in what is being said by their shared words and phrases. Some structural grammarians have sought to avoid this problem by inventing an entirely new vocabulary, but this has not reduced the confusion. In this report, the reader is requested, therefore, to observe the definitions of terms and not to impose classical inferences on them beyond the limits of the definitions.

The goal of the structural grammarian is to formalize a general theory of structural grammar which will be applicable to all languages, or at least to all languages of interest to the linguist. Additionally, he is interested in formalizing general principles for discovering the structural grammar of a given language. No universal theory yet exists, but grammars have been developed which approximate the structure of certain natural languages. Present theories only partially meet requirements for a general theory.

The minimum criterion for any acceptable grammar of a language is that the grammar be weakly equivalent to the implicit grammar of a native speaker of the language, preferably an educated speaker. Chomsky[4] calls two grammars *weakly* equivalent if they generate the same set of sentences from the same initial vocabulary, or, from an analytical viewpoint, if they classify the same strings as sentences and non-sentences. He calls two grammars *strongly* equivalent if there is an isomorphism between the structural diagrams which each grammar associates with sentences. The following descriptions of the various types of grammar have been adapted from an excellent summary by Bobrow.[42]

3.2.1 Dependency Grammars

Dependency grammars such as that developed by Hays[43,44] are, conceptually, the simplest type. A sentence is viewed as being constructed from a hierarchy of dependency structures in which all words are related to the sentence by a dependence on another word, except for an original word (usually the main verb). For example, adjectives depend upon the nouns they modify; nouns depend on verbs as subjects and objects, and on prepositions as objects; adverbs and auxiliaries depend on verbs.

The phrase "the boy" is made up of two elements with *the* dependent on *boy*. In the phrase "at home," *home* is dependent on the preposition *at* to connect it to the rest of the sentence. Figure 1-4 is a graphic representation of the syntactic structures associated with some strings by a dependency grammar. The structures are downward branching trees with each node of the tree labelled with a word. A word is dependent on the word immediately above it in the tree. This type of grammar is good for graphically illustrating deep structural relationships in a sentence, but it does not lend itself well to identifying the various types of dependencies nor to formal notation. Therefore, it is not included among those grammars considered "transformational."

3.2.2 Categorical Grammars

The study of categorical grammars was begun by Ajdukiewicz[45] and continued by Bar-Hillel[46] and Lambeck.[47] The purpose of these grammars is to provide a computational approach to syntactic analysis. The immediate constituent grammars require two independent dictionary look-up operations which can require significant time on a computer, especially when the list of grammar rules is long. Computational techniques would reduce the time required for computer analysis.

The work is based on the following concept. In classical physics, the dimensions of the two sides of an equation can be used to determine its grammatical correctness. Properties similar to dimensions can be assigned to the various grammatical categories of language which enables a similar computation of grammatical correctness.

For example, Bar-Hillel assigns the grammatical code "n" to a noun, and the code $\frac{\text{"n"}}{[n]}$ to an adjective. Thus an adjective-noun string is represented as

$$\frac{n}{[n]} \cdot n$$

By performing a "quasi-arithmetic" cancellation from the right, the code for the string is computed to be

$$\frac{n}{[n]} \cdot n = n$$

This computation essentially states that an adjective-noun string can be treated in the same way as the original noun. As another example, an intransitive verb such as *sleep* in "children sleep" is given the code $\frac{\text{"s"}}{(n)}$. The string "children sleep" is coded as

$$n \cdot \frac{s}{(n)} = s$$

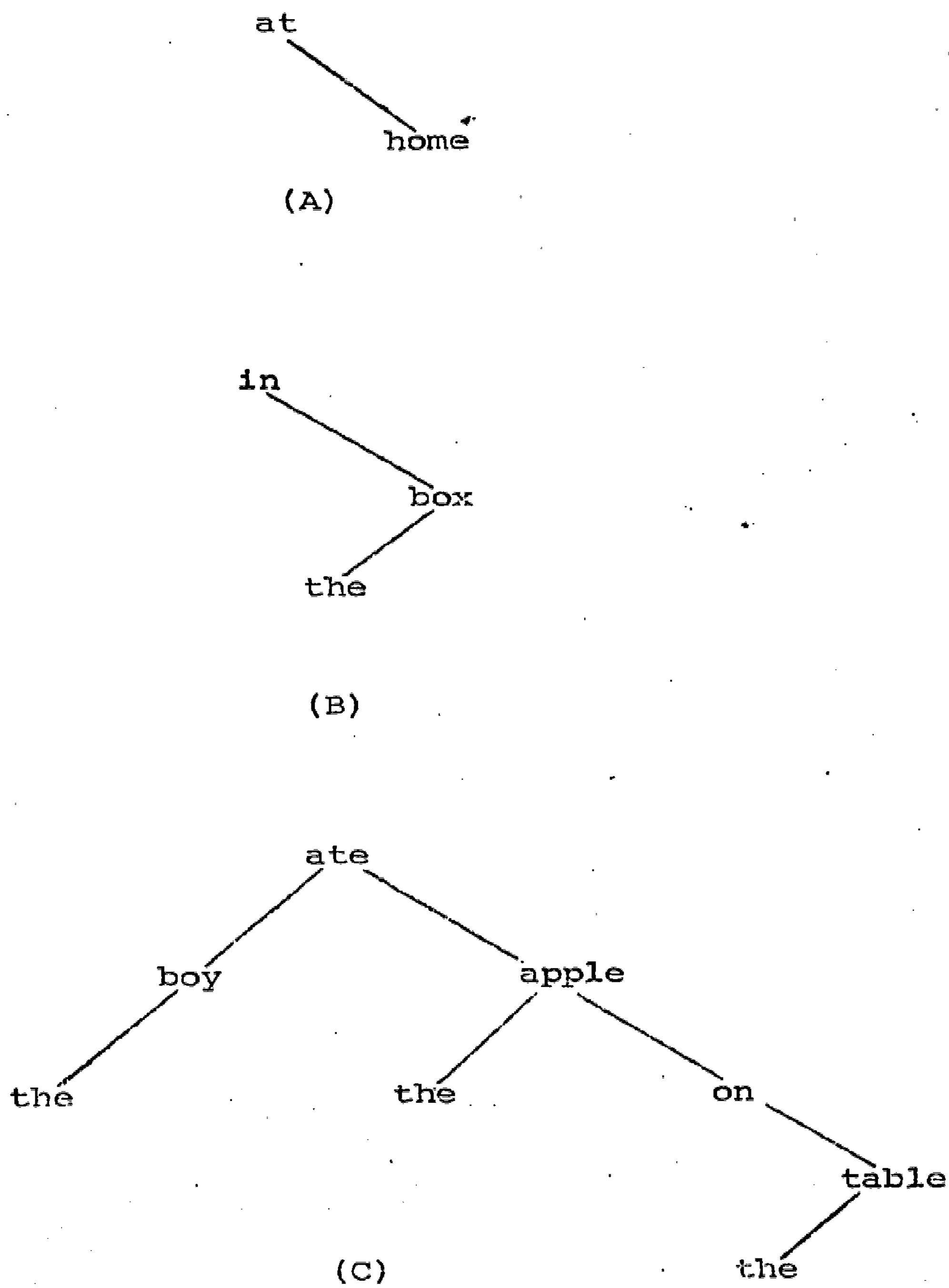


Figure 1-4. Dependency Grammar

where cancellation is performed from the left. Obviously, this coding cannot be distributive since the string "tired children" is permissible, but "children tired" is not. Therefore the brackets [] indicate cancellations permissible from the right and the parentheses ( ) indicate cancellations permissible from the left. The string "tired children sleep" is coded as

$$\frac{n}{[n]} \cdot n \cdot \frac{s}{(n)}$$

By performing cancellations first from the right, the computation produces

$$\frac{n}{[n]} \cdot n \cdot \frac{s}{(n)} = n \cdot \frac{s}{(n)}$$

By then performing cancellations from the left, the computation produces

$$n \cdot \frac{s}{(n)} = s$$

which indicates the string forms a grammatical sentence.

There are many problems implicit in dealing with such string markers which the simple illustrations do not reveal. These problems have been further investigated, but very little has been done to develop an extensive grammar of this form for English. Categorical grammars are suitable for dealing with sequences in a sentences, but not with many other features of a language. Therefore, they are not included among the "transformational" grammars.

3.2.3 Phrase-Structure Grammars

A phrase-structure grammar is a formalization of "immediate constituent analysis" which was first introduced by Leonard Bloomfield.[48] The basic premise of immediate constituent analysis is that contiguous substrings of a sentence are syntactically related. Chomsky[4] calls this type of grammar a context-free phrase-structure grammar. This grammar groups the words of a sentence into phrases which are further subdivided into smaller constituent phrases, the process continuing until the ultimate constituents are reached. A phrase-structure grammar is defined as a finite vocabulary (list of symbols), a finite set of initial symbols, and a finite set of rules. The set of initial symbols provides a list of starting points for the grammar, and the symbols represent the most general constituent members of the grammar. For example, one of the symbols "SENTENCE," "QUESTION," or "COND-SEN " may be used as a starting symbol for the grammar to generate a *simple sentence*, a *question*, or a *conditional sentence*, respectively.

The rules are of the form: X = Y, where X and Y are sequences of symbols. Each rule is to be interpreted as the instruction, "replace X with Y." For example, if a given rule is written

(a) SENTENCE = NP + VP

it means that the symbol "SENTENCE" is to be replaced by the symbols "NP + VP." If the grammar originally selected from among the list of initial symbols the symbol "SENTENCE," it has determined that it will construct a *simple sentence* rather than a question or some other unit. If it then selects rule (a) to operate on the initial symbol, it has determined that the sentence to be constructed will contain a noun phrase (NP) followed by a verb phrase (VP). Therefore, it replaces the symbol "SENTENCE" with the sequence of symbols "NP + VP." It has thus moved from a very general constituent to a sequence of more specific constituents.

The grammar continues to move from the general to the specific by a sequence of rules until a terminal sequence is obtained. A terminal sequence is a sequence of terminal symbols each of which has no further applicable rule: each terminal symbol is a word in the language of the grammar.

The rules of the grammar preferably are applied in a specific order and are designated either as *obligatory* rules which must be applied when reached in the sequence, or as *optional* rules which need not be applied.

Figure 1-5 is a tree diagram of the phrase structure of the sentence "the boy ate the apple." A tree diagram is helpful for illustrating the rank of the symbols and their interrelationship, but it does not lend itself to being presented in formal terms. A system of initial symbols and rules is much better for formal presentation. An example of a phrase-structure grammar is given in Section 4.

Basically, phrase-structure grammar is more powerful than a finite-state grammar. However, it has two important weaknesses which, according to Chomsky, limit its usefulness for English and perhaps for other languages as well:

1. It has no place for discontinuous elements--it does not allow for phrases that may be interrupted or divided in a noncontinuous fashion.

2. It allows for no knowledge of the "history of derivation" of a string--it does not allow for keeping track of what happened in previous rules in addition to the rule presently operating.

Although it is generally accepted that English can be described by phrase structure, such description is lengthy and cumbersome. However, as shown later, these limitations can be rectified by applying proper

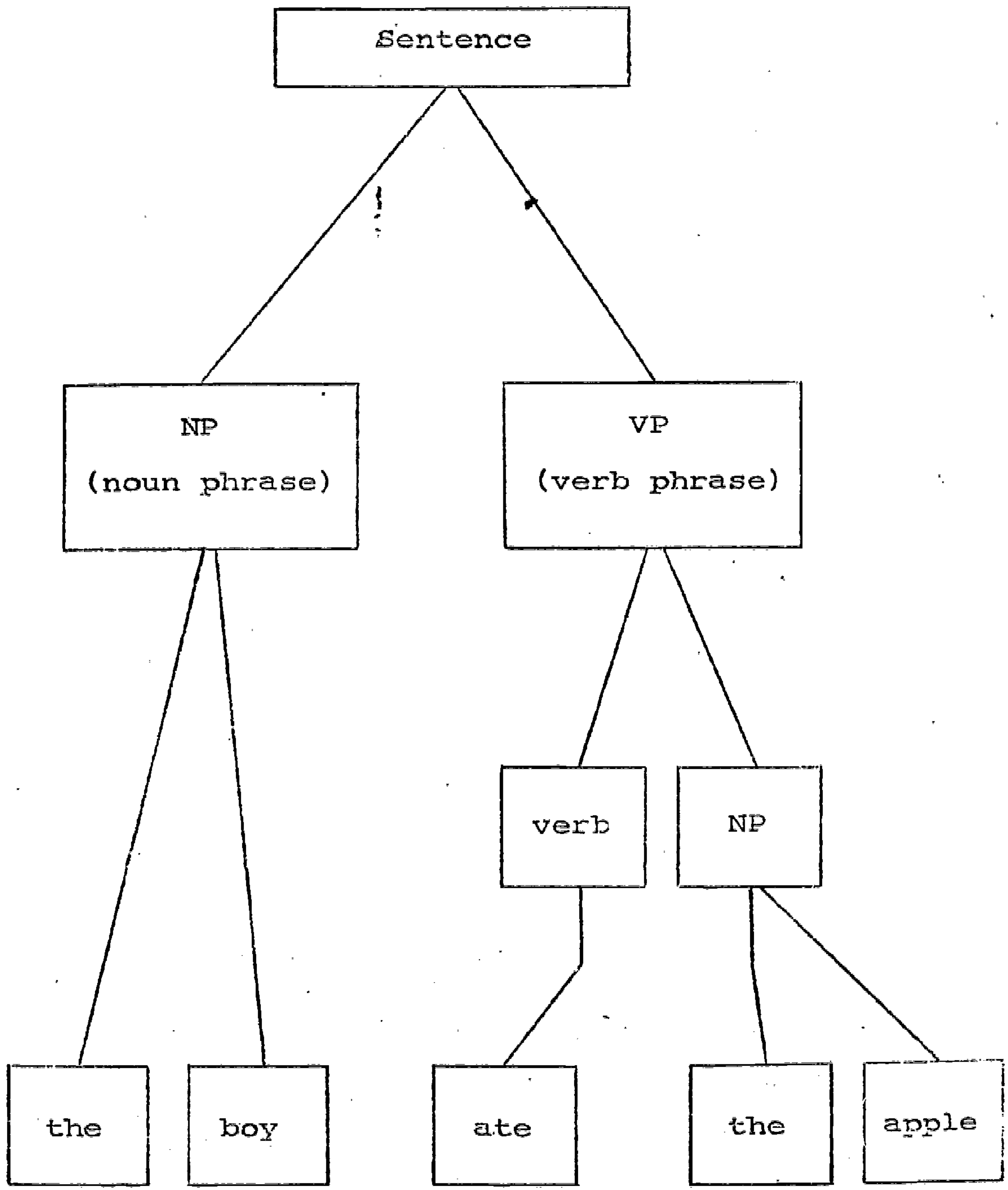


Figure 1-5. Tree Diagram of Phrase-Structure Grammar

restrictions to the notational system of phrase structure. This is variously accomplished in several of the grammars that follow.

3.2.4 Predictive Syntactic Analysis

Predictive syntactic analysis is based on a very restricted form of an immediate constituent (phrase-structure) grammar. Most immediate constituent parsing techniques require many passes over the input string and often consider internal substructures of the string before constituents containing the initial words. A predictive parser analyzes a sentence in one scan of the words from left to right.

Predictive analysis is based on the assumption that when a word of a sentence is given, certain words are expected to follow. For example, if the word "the" is given, one expects that later in the sentence a noun will appear. Thus the prediction of a noun can be made. An alternative expectation would be an adjective. Further, given the constituent ingredients of a subject, one expects a verb to follow. Following this procedure, a list of predictions can be made of the possible words expected to follow a given syntactic situation. By possessing a complete list of predictions, the grammar is equipped to parse sentences in one pass.

The first work on predictive syntactic analysis was by Ida Rhodes[9,10] for a Russian parsing program. The most extensive grammar for English was developed at Harvard by Kuno and Oettinger.[11,12] Robert Lindsay[13] has also written a parsing program using predictive analysis techniques. However, Lindsay is interested in the problem of extracting information from text and answering questions rather than in translation.

3.2.5 Transformational Grammar

This approach was first introduced by Zellig Harris[2] as the result of an empirical study of the structures of language. It was further developed by his student Noam Chomsky.[4] This theory presents the concept of language as having a simple set of "kernel sentences" which are described by phrase structure and which may be operated on by rules of transformation to derive more complex sentences of the phrase-structure type. For example, a kernel sentence should be a simple declarative such as "the boy ate the apple." This simple sentence could be transformed into its equivalent passive form "the apple was eaten by the boy." Chomsky[3] points out that the grammar of English is simplified if phrase-structure description is limited to a kernel of simple sentences from which others are formed by one or more transformations.

Chomsky proposes that the phrase-structure rules be rewritten as

$$Z\ X\ W = Z\ Y\ W$$

where Z and W are the context of the single symbol X, and Y may be strings of one or more symbols. This forms a context-sensitive phrase-structure grammar which operates on a simple set of "kernel sentences."

Transformational grammars permit the basic phrase-structure grammar to be simpler. They account for the relationship between a simple sentence and its derived forms, such as the relationship between the active and passive, and the relationship of the sentence

the boy ran away

and the phrase

the boy who ran away.

They also account for the relationship between such phrases as "the dog is running" and "the running dog." In addition, if certain "semantic" restrictions are to be included in the grammar, they need only be imposed on the phrase-structure rules and written only once.

Transformational grammars have heretofore been considered difficult to implement on a computer, but Friedman has recently developed a computer model of such grammars [49]

The following simplified example illustrates a transformational grammar.

Let the transformational grammar G_t be defined as a phrase-structure grammar G_p which defines deep structure, and a set of transformations T which defines rearrangements of the elements of G_p.

Let the phrase-structure grammar G_p be defined as a set of symbols S and set of replacement rules R on the symbols of the form

$$A = B + C + D$$

which is interpreted "replace A with B + C + D."

Let the transformations T be of the form

$t_1 : 1 + 2 + 3 \rightarrow 3 + 2 + E + 1$

which is interpreted "for the given rule, rearrange the sequence of the elements from 1 + 2 + 3 to 3 + 2 + E + 1, inserting E as indicated."

The grammar then is defined as follows:

$G_t : G_p, T$

$G_p : S, R$

$S : A, D, N_o, N_s, N_1, N_2, SEN, P, V$

$R : SEN = N_s + V + N_o$

$N_s = D + N_1$

$N_o = D + N_2$

D = the

N_1 = boy

N_2 = apple

P = by

V = ate

W = who

$T : t_1 : 1 + 2 + 3 \rightarrow 3 + 2\ (pas) + P + 1$

$t_2 : 1 + 2 + 3 \rightarrow 1 + W + 2 + 3$

Beginning with symbol SEN, the phrase-structure grammar G_p generates the following derivation of a deep-structure "kernel sentence."

SEN

$N_s + V + N_o$

$D + N_1 + ate + D + N_2$

the boy ate the apple.

Transformation t_1 could be applied to the derivation to produce the surface structure of the passive as follows:

SEN

$N_s + V + N_o$

t_1: $1 + 2 + 3$

$3 + 2(\text{pas}) + P + 1$

$N_o + V(\text{pas}) + P + N_s$

$D + N_2 + \text{was eaten} + D + N_1$

the apple was eaten by the boy.

Transformation t_2 could be applied to the derivation to produce the surface structure of a relative-clause noun-phrase as follows:

SEN

$N_s + V + N_o$

t_2: $1 + 2 + 3$

$1 + W + 2 + 3$

$N_s + W + V + N_o$

$D + N_1 + \text{who} + \text{ate} + D + N_2$

the boy who ate the apple.

These simple examples illustrate how different surface structures are derived from the same "kernel sentence" by means of transformations. The meaning is contained in the kernel sentence, whereas different shades of meaning are produced in the surface structure by means of transformations. In reality, transformational grammar also may be viewed as a highly restricted form of an immediate constituent grammar, part of the restrictions of which are written in a second notational system (called a "transformational" system of notation).

3.2.6 Phrase-Structure Grammar with Complex Constituents

Harmon[14] has written a generative phrase-structure grammar

without transformation rules which he claims to have all the advantages of transformational grammars. Additional power is introduced into phrase-structure grammar by the use of complex symbols for syntactic markers. An example of a complex syntactic marker that might be used is

"SENT/SUBJ ABSTR, OBJ ANIM"

This is interpreted as a marker for a sentence which has an abstract subject and an animate object. The designators following the "/" are the subscripts of the symbol. The rewrite rules of the grammar may operate on the symbol, on its subscripts, or on both.

This grammar permits "semantic" restrictions to be accounted for at a high hierarchial level. In addition, both passive and active constructions are generated from one sentence specification, thus accounting for their close relationship. The length of this grammar is approximately the same as a transformational grammar. It has the advantage of using an unordered set of rules, which is untrue of transformational grammar. Thus the use of complex symbols seems to provide all the advantages of a transformational grammar. It must be kept in mind, however, that the transformational grammar has more generative power, but this facility may never be required in practice.

3.2.7 String Transformational Grammars

Zellig Harris[5] and his associates at the University of Pennsylvania have developed a grammar which is intermediate between a phrase-structure grammar (immediate constituent analysis) and a transformational grammar. The basic assumption of string transformational grammars is that a sentence has one "center" which is an elementary sentence. The "center" represents the basic structure of the sentence. Additional words in the sentence are considered as adjuncts to these basic words or to structures within the sentence. Analysis of a sentence consists of identifying the center of the sentence and adjoining the remaining words to the proper elements of the sentence. For example, Harris gives the following analysis:

"Today, automatic trucks from the
factory which we just visited carry
coal up the sharp incline."

Trucks carry coal is the center, elementary sentence; *today* is an adjunct to the left of the elementary sentence; *automatic* is an adjunct to the left of *trucks; just* is an adjunct to the left of *visited,* and so on.

Joshi,[6] an associate of Harris at the University of Pennsylvania, has done later work on string analysis which tends to make its results

more like those of transformational analysis. He resolves a sentence into a number of kernel sentences so that each main verb in the sentence is part of its own kernel.

Naomi Sager,[7,8] another associate of Harris, has directed the programming of a predictive procedure for string analysis. The procedure is similar to phrase-structure predictive analysis and it is written to find all possible string analyses of a sentence.

3.3 Comparison of Grammars

The various types of grammars presented above should not be considered as competing theories. Each type of grammar uses a different property of sentences as a basis for describing the whole of a language, and each has advantages and disadvantages resulting from the choice of the selected property. Actually, sentences exhibit all these properties simultaneously, and when one property is used as the basis for describing a language, the effects of the other properties become restrictions on the chosen property. Thus the question as to which grammar is best becomes meaningless. Granted sufficient restrictions, each type can describe a language equally well. That is why Predictive Syntactic Analysis Grammars, String Analysis Grammars, and Complex-Constituent Phrase-Structure Grammars are all considered "transformational-type" grammars. They all (including transformational grammar) may be considered various forms of highly restricted phrase-structure grammars. A more meaningful question is which grammar is best for a given application. Problems of mechanization and considerations of desired results enter here. A potential user should consider the various types in light of his particular needs and select the type best suited for his requirements. For this work, a *phrase-structure grammar with complex constituents* was selected. Some reasons for this choice are given later.

4. COMPLEX-CONSTITUENT PHRASE-STRUCTURE GRAMMARS

This section provides a formal description of complex-constituent phrase-structure grammar which is the theoretical linguistic model used in this project. First, a formal description is given of a simple phrase-structure grammar, that is, without complex constituents. Then the limitations of this simple form of the grammar which render it inadequate for natural languages such as English and Hebrew are discussed, and it is shown that the use of complex constituents (i.e., subscripted symbols) provides a notational mechanism for imposing the restraints necessary for overcoming these problems. Finally, the general requirements for applying complex-constituent phrase-structure grammar to Semitic languages are outlined.

4.1 Description

4.1.1 Simple Phrase-Structure Grammars

Phrase-structure grammar is a formalization[3,4] of "immediate constituent analysis" which was first introduced by Bloomfield.[48] The basic premise of immediate constituent analysis is that contiguous substrings of a sentence are syntactically (structurally) related.* That is, the deep structure of the information is encoded in the contiguous sequential order of symbols and groups of symbols. Languages for which the basic premise is true are classified as phrase-structure languages. Such languages can be used to communicate messages for information systems with structural patterns that can be mapped after the fashion of Figure 3-1(c). They are inadequate for more complex structural patterns.

Phrase-structure grammars may be considered as information-processing systems that arrange the sequence of symbols and groups of symbols of a message so as to encode the deep structure of the information. They are represented by the following system of notation.

Given a phrase-structure language L with vocabulary V containing a symbol for each information unit, valid statements (sentences) in L are synthesized (encoded) by a generative phrase-structure grammar G_L^g which consists of a set of symbols Ψ_L^g and a set of ordered replacement rules Ω_L^g on the symbols. Valid statements in L are analyzed (decoded) by an analytic phrase-structure grammar G_L^a which consists of a set of symbols Ψ_L^a and a set of ordered replacement rules Ω_L^a on the symbols. For nonambiguous languages, G_L^a is a mirror image of G_L^g.

Consider a generative phrase-structure grammar G_L^g. The set of symbols consists of (1) a set of initial symbols Ψ_1 which are used to initiate sentences, (2) a set of intermediate symbols ψ_2 which define deep-structure relationships,† and (3) a set of terminal symbols $\psi 3$, which are identical with V. The set of replacement rules transforms the structural information to sequential position‡ and is of the form:

---

*See discussion in Section 3.2.3.

†From the viewpoint of surface structure, the symbols represent phrases (groups of words) and smaller constituent phrases (sub-groups of words) that make up a sentence in the language. From the viewpoint of deep structure, the symbols represent the various types of structural relationships that may be made with the information in the associated information system.

‡From the viewpoint of surface structure, the rules define a phrase as to content and sequential order. From the viewpoint of deep structure, the rules define hierarchical dependency of the various types of structural relationships; the deeper the structural relationship, the higher the hierarchical level.

(i) A = B + C

(ii) B = A + C

(iii) C = B + A

where A, B, C and D are symbols of the grammar. The sign = is interpreted "replace the symbol on the left of = by the symbols on the right." The sign + links symbols in a sequential series. The Roman numerals define the order of the rules.

The grammar works as follows: beginning with an initial symbol, replacement rules are applied according to hierarchical order, thus producing a new sequence of symbols. The process is repeated until only terminal symbols remain. Alternative choices produce variations in the surface structuce of the sentence being generated.

4.1.2 Illustration of a Simple Phrase-Structure Grammar

The following is an example of a simple phrase-structure grammar. Given the artifical language L with the vocabulary

V = {the, boy, girl, children, bought, ate, hid, apple, pie, candy} (1)

the grammar G_L is defined as

G_L : $\{\Psi_L, \Omega_L\}$ (2)

ψ_L : $\{\psi_1, \psi_2, \psi_3\}$ (3)

ψ_1 : SENTENCE

ψ_2 : {NP_1, NP_2, VP, VERB, $NOUN_1$, $NOUN_2$}

ψ_3 : {the, boy, girl, children, bought, ate, hid, apple, candy}

Ω_L : (4)

(i)	SENTENCE	=	NP_1 + VP
(ii)	VP	=	VERB + NP_2
(iii)	NP_1	=	the + $NOUN_1$
(iv)	NP_2	=	the + $NOUN_2$
(v)	VERB	=	ate/bought/hid
(vi)	$NOUN_1$	=	girl/boy/children
(vii)	$NOUN_2$	=	pie/apple/candy

37

The grammar begins with the initial symbol

SENTENCE (step 1)

It then applies each rule in sequence as indicated by the sequence number in the parentheses. Rule (i) says to replace SENTENCE with "NP_1 + VP," which leaves

NP_1 + VP (step 2)

Rule (ii) says to replace VP with "VERB + NP_2," which leaves

NP_1 + VERB + NP_2 (step 3)

Rule (iii) says to replace NP_1 with "the + $noun_1$," which leaves

the + $NOUN_1$ + VERB + NP_2 (step 4)

Rule (iv) says to replace NP_2 with "the + $NOUN_2$," which leaves

the + $NOUN_1$ + VERB + the + $NOUN_2$ (step 5)

Rule (v) says to replace VERB with either "ate," "bought," or "hid"; select "ate," which leaves

the + $NOUN_1$ + ate + the + $NOUN_2$ (step 6)

Rule (vi) says to replace $NOUN_1$ with either "girl", "boy" or "children"; select "boy," which leaves

the boy ate the + $NOUN_2$ (step 7)

Rule (vii) says to replace $NOUN_2$ with either "pie," "apple," or "candy"; select "apple," which leaves

the boy ate the apple (step 8)

Since all symbols are terminal symbols, the grammar can proceed no further; the desired sentence is constructed (without punctuation).

The above example demonstrates how the grammar is used to generate or synthesize a sentence. By selecting the various other optional choices, the grammar will generate 27 different sentences.

However, the same grammar may be used in reverse to analyze a sentence. Assume the same grammar as before, and assume the terminal sequence of symbols, "the boy ate the apple," which is to be analyzed to determine whether or not it is a valid sequence in the given grammar. The analysis procedure begins with the terminal sequence

$$\text{the boy ate the apple} \qquad \text{(step 1)}$$

and applies the rules in reverse sequence. If the grammar successfully arrives at an initial symbol, it has determined that the sequence of symbols is valid. Not only is the sequence reversed, but also the interpretation of the rules. For the analysis procedure, the rule X = Y is interpreted as the instruction, "replace Y with X." Following through on the example, Rule (vii) says to replace "apple" with $NOUN_2$, and Rule (vi) says to replace "boy" with $NOUN_1$, which leaves

$$\text{the} + NOUN_1 + \text{ate the} + NOUN_2 \qquad \text{(steps 2 \& 3)}$$

Rule (v) says to replace "ate" with VERB which leaves

$$\text{the} + NOUN_1 + VERB + \text{the} + NOUN_2 \qquad \text{(step 4)}$$

Rules (iv) and (iii) say to replace "the + NOUN2" with "NP_2" and to replace "the + $NOUN_1$" with "NP_1" respectively, which leaves

$$NP_1 + VERB + NP_2 \qquad \text{(steps 5 \& 6)}$$

Rule (ii) says to replace "VERB + NP_2" with VP, which leaves

$$NP_1 + VP \qquad \text{(step 7)}$$

Rule (i) says to replace "NP_1 + VP" with SENTENCE, which leaves

$$SENTENCE \qquad \text{(step 8)}$$

This symbol is an initial symbol which indicates that the sequence of terminal symbols under analysis is a valid sequence in the grammar.

The two examples demonstrate how a phrase-structure grammar may be used for the synthesis or analysis of sentences in a language. The examples are very simple and do not cover complexities which may be encountered in natural languages.

4.1.3 Limitations of Simple Phrase-Structure Grammars

The simple phrase-structure grammars defined and illustrated in the previous section are limited to syntax only, that is, to encoding deep-structure information into sequential relations only. However, because natural languages use a combination of sequential and symbolic encoding, simple phrase-structure grammars (as well as any other type confined to sequential encoding only) are inadequate for these extra features of natural languages. Some of their inadequacies have been mentioned before. This section discusses the inadequacies in detail and shows what modifications of the grammars are required to account for these extra features of natural languages.

4.1.3.1 Lack of Option Notation

Languages exhibit the characteristic that various types of phrases may serve the same syntactic function in a sentence. For example, in the sentences

(a) the meeting was *a victory party*

(b) the meeting was *good*

(c) the meeting was *in Town Hall*

(c) the meeting was *at noon*

the phrase that completes the meaning of the copula is a different type for each one. In (a) it is a noun phrase N_p, in (b) an adjective phrase A_p, in (c) an adverb phrase of space D_{ps}, and in (d) an adverb phrase of time D_{pt}. The adverb phrases of (c) and (d) may be considered as subclasses of a general adverb phrase D_p.

The rules of simple phrase-structure for defining these sentences would be

$$\begin{aligned} &(a) \quad S_d = N_{sp} + V_1 + N_p \\ &(b) \quad S_d = N_{sp} + V_1 + A_p \\ &(c,d) \quad S_d = N_{sp} + V_1 + D_p \end{aligned} \tag{5}$$

If the notation permitted optional choices, the three rules could be combined in to one such as

$$S_d = N_{sp} + V_1 + \begin{Bmatrix} N_p \\ A_p \\ D_p \end{Bmatrix} \tag{6}$$

Or, to make things simpler, a symbol for a copulative phrase N_{px} could be provided and defined by a new rule such as

$$\begin{aligned} &(a) \quad S_d = N_{sp} + V_1 + N_{px} \\ &(b) \quad N_{px} = \begin{Bmatrix} N_p \\ A_p \\ D_p \end{Bmatrix} \end{aligned} \tag{7}$$

The rule of (7a) now defines the structure of sentences of *definition* S_d, where the subject phrase N_{sp} is the thing being defined, V_1 is the copula *is*, and the copulative phrase N_{px} defines the kind of *definition* being imposed on N_{sp}. Thus, if N_{sp} is being defined as to *name dimension*, then

$$N_{px} = N_p$$

and the sentences says

$$N(N_{sp}) = N_p$$

which means "N_{sp} possesses a *name* which is N_p."

If N_{sp} is being defined as to *semantic dimension*,* then

$$N_{px} = A_p$$

and the sentence says

$$A(N_{sp}) = A_p$$

which means "N_{sp} possesses the *semantic dimension A*, the value of which is A_p. If N_{sp} is being defined as to the *space-time dimension* then

$$N_{px} = D_p$$

and the sentence says

$$D(N_{sp}) = D_p$$

which means "N_{sp} possesses a *space-time dimension D*, the value of which is D_p." Thus the symbol for the copulative phrase N_{px}, the name of which defines its syntactic function, is found to correspond to a linguistic feature which is called *definition* herein.

When the same process of combining simple phrase-structure rules like (5) into rules like (7) is applied repeatedly to the grammar, the number of rules is reduced, and the resultant set of nonterminal symbols is found to map the relationship of syntactic functions to their corresponding linguistic features. Thus, there will be intermediate symbols that uniquely correspond to such linguistic features as *voice*, *mood*, *tense*, *nominalization*, *quantification*, *qualification*, and so forth. Likewise, the optional choices defined by the rules on a given symbol will correspond to the different values that the associated linguistic feature may assume. For example, the rule on the symbol that corresponds to the feature *voice* would have options that correspond to the values that *voice* may assume; namely, *active*, *passive*, and *reflexive*.

Providing phrase-structure notation with this power of optional choice gives it computing capability equivalent to that provided by a

*The term *semantic dimension* is used in the sense of *adjectival quality*, so that the adjective *small* is considered a value on the scale of the semantic dimension *size*.

certain class of transformations (called *option* transformations herein), but without the use of a second "transformational" system of notation. It also enables the grammar to explain the common deep-structure relationships that exist between such forms as the *active* and *passive* voices of sentences by showing that they originate from different options of the same symbol. In addition, it reduces the number of rules in the phrase-structure grammar.

In using a phrase-structure grammar to generate sentences, the optional choices available to grammar rules, such as in (7b), alter the information content of the resultant sentence. If a specific message is to be encoded into a sentence of the language, then the choices may not be made on a random basis, but they must be governed by the information content of the given message. For example, the rule of (7b) means that N_{px} may be replaced by either N_p, A_p, or D_p. Actually, the choice depends on the message being encoded, but there is no notation for imposing this choice on the grammar rule for a given application. What the notation of the rule needs is a subscript for the symbol by means of which the choice may be imposed. Thus (7b) must be rewritten

$$N_{px(c)} = \begin{Bmatrix} N_p \\ A_p \\ D_p \end{Bmatrix}^{c}, \quad c = 1,2,3 \tag{8}$$

which means: c is assigned the value 1, 2, or 3 and then

$$N_{px(1)} = N_p$$
$$N_{px(2)} = A_p$$
$$N_{px(3)} = D_p$$

Since the choice of the value for c depends on the information content (I) of the message, that is,

$$c = \Phi(I) \tag{9}$$

the rule should be written

$$N_{px(c)} = \begin{Bmatrix} N_p \\ A_p \\ D_p \end{Bmatrix}^{c}, \quad c = \Phi(I) \tag{10}$$

thus relating the operation of the rules to the message being encoded.

However, in addition to this, the grammar provided no means for the computation of (c), that is, for relating the information (I) of a message to corresponding options of the rules. This deficiency is met by providing the grammar with a set of operational functions Φ which define the value of c as a function of information (I) for each rule. In addition, the grammar must have the facility for defining the content of the information (I) of a given message to be encoded. This is provided by adding (I) to the grammar, where (I) represents the input data required to define the information content of a given message.

Further consideration reveals that the value of subscript c, as computed by the operational function, is dependent only on the information unique to a symbol as it relates to the past history of the derivation. However, the notation of phrase-structure does not provide for recording deep-structure dependencies (derivational history). Thus, there is not enough data to retrieve the information unique to a given symbol. The minimum required to retrieve the information unique to a given symbol is one index number q, values of which are assigned so that the q-th symbol of the derivation and the q-th information unit of (I) are in proper correspondence.

Thus, the grammar must be defined as

$$G_L: \{\psi_L, \Omega_L, \Phi_L, I\} \tag{11}$$

where

$$\Phi_L: \{\phi_A(I,q), \phi_B(I,q), \ldots\} \tag{12}$$

and the rules are of the form (10).

With the notation of phrase-structure grammars thus modified, it is provided with the capabilities of "option transformations" without the use of a second system of notation.

4.1.3.2 Lack of Universal Rules

Natural languages employ two schemes for grouping words. The first scheme involves arranging words in groups that can be uniquely identified as "phrases." This is the scheme that simple phrase-structure notation is designed to handle. The second scheme involves grouping patterns that are common to many different phrases and thus have a "universal"* application. Examples of universal grouping patterns are:

*The term "universal" is used in the sense that the patterns apply to many different symbols of the grammar but not necessarily all.

1. compounding--joining like symbols with conjunctions
2. negation--attaching negatives to various symbols
3. determination--attaching the definite or indefinite article to various symbols
4. deletion--omitting optional symbols.

Simple phrase-structure notation is inefficient for this second scheme because it requires a separate statement of rules which have the same form but different symbols. For example, given the rules:

$$\begin{aligned} A &= A + C + A \\ B &= B + C + B \\ D &= D + C + D \end{aligned} \tag{13}$$

all rules have the common form

$$F = F + C + F \tag{14}$$

They differ only by the symbol occupying the position of F. It would be nice if universal rules of this type could be written as (14) is written rather than (13). This improvement can be made by providing the grammar with (1) a variable symbol F--one that stands in place of other symbols, (2) a set of universal rules on F in phrase-structure form, and (3) a set of subscripts that governs the rules. For example, given the rule on F

$$F_{ijf} = \begin{Bmatrix} F + \text{AND} + F \\ F + , + F + \text{AND} + F \end{Bmatrix}^{i}, \quad i = \phi_F(I,j,f), \; f \neq 0 \tag{15}$$

the rule operates on symbol A as though it were written

$$A_{ijf} = \begin{Bmatrix} A + \text{AND} + A \\ A + , + A + \text{AND} + A \end{Bmatrix}^{i}, \quad i = \phi_A(I,j,f), \; f \neq 0 \tag{16}$$

and on symbol B as though it were written

$$B_{ijf} = \begin{Bmatrix} B + \text{AND} + B \\ B + , + B + \text{AND} + B \end{Bmatrix}^{i}, \quad i = \phi_B(I,j,f), \; f \neq 0 \tag{17}$$

and so forth. The rule does not operate if $f=0$.

Providing phrase-structure notation with the power of universal rules greatly reduces the number of rules required by the grammar; at the same time it gives the grammar computing capabilities equivalent to a second class of transformations (called *universal* transformations herein) but without the use of a second "transformational" system of notation.

It also enables the grammar to explain the universal patterns of the language that transcend the bounds of phrases by a few simple rules in phrase-structure notation.

4.1.3.3 Lack of Semantic Restraints

Natural languages usually require agreement between the common inflectional features of words that are structurally related. Thus, for example, in Hebrew the inflection of a verb must agree with that of the subject in number, gender and person, and an adjective must agree with the noun if modifies in number, gender and determination. There are traces of this in English in such cases as *I walk, he walks,* but not **I walks, *he walk.* This feature of language has been referred to as *context sensitivity.* It implies that some rules of the grammar operate on a symbol only in a given environment and thus must be written in the form

$$V + X + W = V + Y + W$$

which means that X in the environment of V and W is replaced by Y, otherwise not. Thus the rule for the previous example would be

$$\left\{\begin{matrix}\text{he}\\ \text{she}\\ \text{it}\end{matrix}\right\} + \text{walk} = \left\{\begin{matrix}\text{he}\\ \text{she}\\ \text{it}\end{matrix}\right\} + \text{walks}$$

Rules of this type are not within the realm of the definition of simple phrase structure. Thus the more powerful "transformations" have been applied to solve this problem. This is a third type of transformation, called *semantic* transformation herein.

However, the problem takes on a different aspect if it is recognized that in English (as in many inflected languages) pronouns and verbs both possess the linguistic features of *number* and *person.* Thus the English personal pronoun is inflected as in Table 1-1, and the English present tense verb *walk* is inflected as in Table 1-2. If the verb possesses the features of *number, person,* and *tense,* then a rule for the previous example would be

$$\text{Walk (sing, third, pres.)} = \text{Walks}$$

which is within the realm of phrase structure with subscripted symbols.

The fact that the information is common to both pronoun and verb implies that it was defined at a deeper structural level and supplied to both through information-bearing dependent variables. The problem is that there are no information-bearing variables in the grammar for noting or controlling the mutual concord that exists between elements of a phrase.

Table 1-1

INFLECTION OF ENGLISH PERSONAL PRONOUN

Number	Gender	Person	Subject Pers. Pro.	Object Pers. Pro.
sing.	all	first	I	me
pl.	all	first	we	us
all	all	second	you	you
sing.	masc.	third	he	him
sing.	fem.	third	she	her
sing.	neut.	third	it	it
pl.	all	third	they	them

Table 1-2

INFLECTION OF ENGLISH PRESENT TENSE VERB *WALK*

Number	Gender	Person	Verb
all	all	first	walk
all	all	second	walk
sing.	all	third	walks
pl.	all	third	walk

46

The solution to this problem is to provide a set of information-bearing variables which amounts to imposing semantic restraints on the grammar.

For example, suppose the grammar is provided with the following semantic subscripts:

d = determination

n - number

g - gender

p - person

t - time

The rules of the grammar may then be written to distribute the semantic data properly so as to provide the required concord. Suppose the grammar, in the simple notation, has the following rules:

Start: S (18)

S = NS + VP

NS = T + NP

NP = N + AP

VP = V + NO

NO = T + NP

AP = T + A

where the symbols mean

S: sentence

NS: subject phrase

VP: verb phrase

T: article

NP: noun phrase

N: noun

AP: adjective phrase

V: verb

NO: object phrase

A: adjective

Semantic restraints similar to those of Hebrew may be applied to the rules as follows:*

$$\text{Start:}\quad S_{NGP} \tag{19}$$

$$S_{ngp} = NS_{Dngp} + VP_{ngpT}$$

$$NS_{dngp} = T_{d} + NP_{dngp}$$

$$NP_{dngp} = N_{ngp} + AP_{dng}$$

$$VP_{ngpt} = V_{ngpt} + NO_{DNGP}$$

$$NO_{dngp} = T_{d} + NP_{dngp}$$

$$AP_{dng} = T_{d} + A_{ng}$$

where the lower-case subscripts identify dependent variables, and the upper-case subscripts identify independent variables. The values of the independent variables are defined by input data from the information system. The values of the dependent variables have been defined previously, and the rules govern the downward distribution of these data among the constituent elements of a given phrase.

The semantic subscripts, then, are information-bearing variables that enable the grammar to collect information throughout the various stages of the derivation and to distribute it downward as required to the smaller constituent phrases at subsequent stages. These information variables can include information that does not enter into the consideration of concord, such as the root, stem, and inflection of individual words.

The use of semantic restraints on the grammar can be extended to any degree required. However, there is a practical limit. The dream of producing an ideal system of semantic restraints, one that will limit a grammar to the generation of meaningful sentences only, is a vain illusion based on the erroneous assumption that a language is identical with the information system it services, and that it is possible to produce a mathematical model that completely defines meaningfulness. It is sufficient to require a grammar to generate only grammatical (correctly encoded) sentences and to require the information system to define meaning. This implies that the grammar will have sufficient semantic restraints, for example, to require an object for a transitive verb in the active voice, but not in the passive voice. It further implies that the grammar will not be able to evaluate the meaningfulness of a specific subject-verb-object combination. On this basis, we can

*Other subscripts discussed in previous material are omitted here for simplicity of illustration.

expect a grammar to have sufficient semantic restraints to avoid sentences such as "breakfast is eaten Mary," but not to avoid sentences such as "Mary frightens sincerity."

Providing phrase-structure notation with semantic subscripts greatly reduces the number of rules required by the grammar, at the same time it gives the grammar computing capabilities equivalent to a third class of transformations (called *semantic* transformations herein), but without the use of a second "transformational" system of notation. The proper use of these subscripts in the rules provides the grammar with a type of *context sensitivity* sufficient for explaining the semantic concord found in natural languages. In addition, the semantic subscripts enable the grammar to explain the *context-sensitiveness* and the semantic restraints of the language within the phrase-structure rules without a second set of "context sensitive" and "semantic" rules.

4.1.4 Complex Constituents Overcome Limitations

In the previous section, the inadequacies of simple phrase structure were examined and the solutions to the problems were outlined. It was shown that by adding certain restraints to the grammar it is made adequate for defining natural languages such as Hebrew (demonstrated in Part II) and for implementation on computers (demonstrated in Parts III and IV). A major feature of the proposed solutions involved the use of symbols with subscripts (i.e., complex constituents) to impose the necessary restraints on the grammar. The solutions provide the grammar with the advantages of transformational grammar without two of its disadvantages: (1) the use of a second "transformation" notation system, and (2) the use of an ordered hierarchy on the set of rules. This is in agreement with the findings of Harmon[14] (see also Section 3.2.6).

Harmon introduced complex constituents to phrase structure by adding syntactic markers to the symbols. An example of such a complex syntactic marker is

"SENT/SUBJ ABSTR, OBJ ANIM"

This is interpreted as a marker for a sentence which has an abstract subject and an animate object. The descriptors following the "/" are subscripts of the symbol. The notation scheme employed herein is briefer than Harmon's, but it accomplishes the same purposes.

4.2 General Requirements

This section describes the general requirements for complex-constituent phrase-structure grammars of Semitic languages. It is based on the experience derived from the development of such a grammar for modern Hebrew and from knowledge of other Semitic languages such as Arabic, Aramaic, Ugaritic, and Akkadian. Future research will surely result in

simplifications and modifications of this basic model. However, this model provides the groundwork for such research, and generalized computer programs based on this model will provide the tools for such research.

A complex-constituent phrase-structure grammar G_L of a Semitic language L consists of (1) a set of symbols Ψ, (2) a set of subscripts Δ on the symbols, (3) a set unordered replacement rules Ω, (4) a set of mapping functions Φ, and (5) an input function I. Thus

$$G_L: \quad \{\Psi, \Delta, \Omega, \Phi, I\} \tag{20}$$

The contents of each of these elements of the grammar is outlined in the sections that follow.

4.2.1 Symbols

The set of symbols Ψ consists of (1) a set of initial symbols ψ_1, (2) a set of intermediate symbols ψ_2, (3) a set of variable symbols ψ_3, and (4) a set of terminal symbols ψ_4. Thus:

$$\Psi: \quad \{\psi_1, \psi_2, \psi_3, \psi_4\} \tag{21}$$

The *initial symbols* ψ_1 stand for completed sentences in the language. They are used to initiate the generation of a sentence by the grammar. The grammar of Hebrew uses only one initial symbol, and that is probably all that is required for other Semitic languages.

The *intermediate symbols* ψ_2 stand for unique groupings of other structurally related symbols—that is, for unique phrases.* A single symbol is assigned to each syntactically significant grouping of words that may occur in the language. The assignment of symbols is made in accordance with the technique outlined in Section 4.1.3.1, so that the symbols also correspond to the various unique linguistic features of the language and the optional choices defined by the rules on a given symbol correspond to the different values that the associated feature may assume. The grammar of Hebrew presently has 72 intermediate symbols. The assignment of symbols for other Semitic languages will vary from this but will follow the general outline.

The *variable symbols* ψ_3 stand for other symbols in the grammar and are used in the "universal" rules of the grammar. The grammar of Hebrew presently has only one variable symbol, and that is probably all that is required for other Semitic languages.

The *terminal symbols* ψ_4 stand for the various classes of words in the language. The classification of the words is based primarily on the syntactic function of the words in the grammar. The grammar for

*See footnotes in Section 4.1.

Hebrew presently has 20 terminal symbols, but there is evidence* that this number should be reduced to 16. This classification will probably be the same for all Semitic languages.

4.2.2 Subscripts

The set of subscripts Δ consists of (1) a set of "pattern" subscripts δ_1, (2) a set of "option" subscripts δ_2, and (3) a set of "semantic" subscripts δ_3. Thus

$$\Delta : \{\delta_1, \delta_2, \delta_3\} \tag{22}$$

The subscripts may be designated by the rules as either (a) independent variables, the values of which are defined by input data, (b) dependent variables, the values of which have been defined at an earlier stage of the derivation, or (3) fixed values.

The *"pattern" subscripts* δ_1 are variables, the values of which are defined by input data and the rules. They are used to govern the application of the "universal" rules of the grammar. The grammar of Hebrew has the following seven pattern subscripts which should be the same for other Semitic languages.**

m -- optional/mandatory

f -- compounding pattern

b -- connective type

k -- number of times compounded

y -- negative/positive

ℓ -- negative class

d -- indefinite/definite.

The *"option" subscripts* δ_2 are variables, the values of which are defined by the operational functions Φ and which are used to govern the alternative choices available to the applicable grammar rules. The grammar of Hebrew has only two "option" subscripts (*c*--symbol class, and *q*--index number) which are all that should be required for other Semitic languages.

The *"semantic" subscripts* δ_3 are information-bearing variables that define certain semantic attributes of the symbols of the grammar. By means of the semantic subscripts, the grammar rules accumulate semantic information and distribute it to the appropriate symbols at lower hierarchical levels; it also uses the semantic subscripts in the operational functions

* Experience has shown that the construct state of nouns, numbers, participles, and infinitives does not require a separate terminal symbol.

**See Section 2.2.1, Part II of this report, for a detailed definition of these and other subscripts.

to serve as restraints on the computations. The grammar of Hebrew has 16 semantic subscripts which should be the same for other Semitic languages. They are:

n -- number

g -- gender

p -- person

r -- prepositional modifier class

a -- verb modifier class

v -- voice

i -- mood

t -- tense

s -- stem

w_1 -- root letter 1

w_2 -- root letter 2

w_3 -- root letter 3

w_4 -- root letter 4

j -- state

h -- feminine noun class

x -- number gender transform.

It should be pointed out that the other sets of subscripts δ_1 and δ_2 are also related to semantic information, but their functions in the grammar are somewhat different. As far as δ_3 is concerned, the specified semantic subscripts are sufficient to limit the grammar to the generation of "grammatical" sentences but not necessarily "meaningful" sentences.*

It must be pointed out that there is no clear distinction between *grammaticalness* and *meaningfulness*, because there is information, and thus meaning, encoded in the syntactic structure of a sentence. The syntactic structure identifies which group of words is the subject, which is the verb, which is the object, which words are modifiers, and so forth. Thus the coarse detail of the information (its gross structure) is encoded by the syntax. The fine detail of meaning is contained in the semantic information encoded in the individual words. If we say a sentence is grammatical but meaningless we mean that the coarse detail of the message is correct (is meaningful), but the fine detail of the message does not correspond to reality.

Meaningfulness is somehow associated with the interrelationships of information units (words) that are possible in the real universe of a

*See discussion in Section 4.1.3.3 for elaboration of this statement.

given information system. So that, for example, in the universe of humans it is possible for John to love Mary, but it is impossible for Mary to frighten sincerity. Thus, in a natural language, it is meaningful to sa

John loves Mary (23)

but it is not meaningful to say

Mary frightens sincerity. (24)

If sufficient semantic restraints are impossed on the grammar, it is conceivable that only meaningful sentences would be generated. However, this implies that a theoretical model of "meaningfulness" has been defined. For artificial languages this is possible, but for natural languages the task is exceedingly complex. Research is being conducted on the subject and numerous theories have been proposed.* However, the subject is not sufficiently understood to go much beyond that which is suggested here at the present time. When the time comes to add more semantic restraints, the notational mechanism is available.

4.2.3 Rules

The set of replacements rules Ω are of the general form

$$A_\delta = \begin{Bmatrix} B_\delta + C_\delta \\ C_\delta + B_\delta \\ D_\delta \end{Bmatrix} c \qquad (25)$$

where

$$c = \phi_A(I, \delta_A)$$

$$\delta = \phi_\omega(I, \delta_A)$$

The interpretation is in accordance with the explanation previously given in Section 4.1 with the following exceptions or additions:

1. The rules are unordered--the use of subscripted symbols enables the rules to impose a natural order on themselves that needs no outside control.

*See listings in various issues of *Language and Automation* and of *Language Research in Progress*, both from Center for Applied Linguistics, Washington.

2. δ_A is the set of subscripts that apply to Symbol A, the left-hand element of (25), δ_B is the set that applies to Symbol B, and so forth.

3. The variables σ, δ_B, δ_C, and δ_D are defined by the operational functions ϕ_A and ϕ_ω, respectively; these functions are defined in the next section.

A rule is written for each nonterminal symbol of the gramma. in such a way that an optional choice is provided for each value that may be assumed by the distinctive linguistic feature associated with the symbol. For each optional choice, the rule defines (1) the content of the phrase in terms of terminal symbols and/or other smaller phrases, (2) the sequential order of the content, (3) the distribution of redundant semantic information throughout the elements of the phrase, and (4) the semantic data of the content that are fixed or that must be defined by input information from the message being encoded. The grammar of Hebrew presently has 76 rules of this type with a total of 179 alternative choices, which average between two and three options per rule. A different set of rules must be written for each of the other Semitic languages, but the general content of each set will be similar to the Hebrew grammar because of common linguistic characteristics.

4.2.4 Operational Functions

The set of operational functions Φ consists of a "subscript" function ϕ_ω and a set of "option" functions ϕ_A, ϕ_B,... (one for each rule of the grammar). Thus,

$$\Phi: \{\phi_\omega, \phi_A, \phi_B, ..\} \tag{26}$$

The *"subscript" function* ϕ_ω is used for defining the values of the subscripts of the right-hand symbols of a rule in terms of input data or in terms of the defined subscript values of the left-hand symbol. Thus, for example, in (25) δ_B is defined as $\phi_\omega(I,\delta_A)$, In Sections 4.1.3.3 and 4.2.2, it was stated that the rules may designate a subscript as either a fixed, dependent, or independent variable. (See these sections for illustrations of the following explanations.) For *fixed* variable subscripts, the rules themselves assign a value to them, and ϕ_ω assigns the value of the corresponding subscript of the left-hand element of the rule. For the *independent variable* subscripts, ϕ_ω assigns the value defined by the input data (I). Function ϕ_ω operates on all the subscripts except σ which is discussed next.

The *"option" functions* ϕ_A, ϕ_B, etc., are used for defining subscript σ for each symbol in the derivational string. This subscript is different from all others in that its value is determined by a different linguistic feature for each symbol, whereas each of the other subscripts has its value determined by the same unique linguistic feature for all

symbols. For example, the value of subscript n is always defined by the linguistic feature *number*, g by *gender*, p by *person*, and so forth. But the value of subscript c may be determined by the linguistic feature *voice* for Symbol A, by *mood* for Symbol B, by *tense* for Symbol C, and so forth. In fact the value of c for a given symbol is determined by that linguistic feature which is uniquely represented by the symbol. So, just as there is one rule for each nonterminal symbol, there is also one "*option*" function for each nonterminal symbol. Thus, for example, in (25) c is defined as $\phi_A(I,\delta_A)$, where ϕ_A is unique for Symbol A.

The "subscript" function ϕ_ω should be the same for all Semitic languages. The "option" functions will be different for each Semitic language, but they will reflect the common linguistic characteristics of the languages.

4.2.5 Input Function

The input functions (I) is the interface between the information system (source of a message) and the grammar (message encoder) of the language (communication medium). It is a catalogue of all the information contained in the sentence (message) being generated (encoded). The catalogue is organized (indexed) so that the functions Φ of the grammar can retrieve the information pertaining to a given symbol of the derivation upon request.

One of the subscripts used by the grammar is a symbol index number (subscript q). Each symbol used in the derivation of a sentence has a unique value assigned to its subscript q, so that it can be referred to as the q-th symbol of the derivation. The information contained in a given sentence to be generated is catalogued in (I) such that the information pertaining to the q-th symbol of the derivation is recorded in the q-th catalogue location.

The problem of *how* the information gets recorded in (I) is of no importance to the grammar, but the fact that it is there is all important. Apart from (I) and its content, the grammar has no criteria for making decisions. One alternative is that the grammar be given the freedom to make arbitrary decisions on a random basis. The result would be sentences that were grammatical but meaningless; or, assuming sufficient semantic restraints, the result would be sequences of unrelated but meaningful sentences. The other alternative is that the grammar be endowed with sentient intelligence. But this is equivalent to incorporating the information system into the grammar and this is out of the question for natural languages.

The problem of *how* the information gets recorded in (I) is very important to the user of the grammar, however. If the user (the information system) is a human, he must use a catalogue guide (input map) to assist him in recording the information in (I). The guide must contain an

inherent image of the grammar that specifies the information required and the sequential order in which it should be recorded (i.e., assignment of values to q). This is the method presently used for the grammar of Hebrew.* The method is complicated and cumbersome, but it is suitable for purposes of education and research. There are indications that the process can be greatly simplified.

If the information system is the output of an analysis grammar of some other language (as in the case of machine-aided translation), then the information must be transferred from the output format of the analysis grammar to the input format of the synthesis grammar. This operation can be performed by a "transfer function." The transfer function must contain an inherent image of the source (analysis) grammar, an inherent image of the target (synthesis) grammar, and a map of the correspondence of their elements. Much of this process can be mechanized. However, experience has shown that, due to ambiguities in the source language and to a lack of complete correspondence between the elements of the grammars, human intervention is required to resolve some of the transfer problems. This explains the use of the term machine-aided translation. At the present, no "transfer function" exists for machine-aided translation either from or to Hebrew.

5. CONCLUSION

It is concluded that several of the different types of structural grammars examined use different properties of sentences as a basis for describing a language; that the other properties become restrictions on the selected basic property; that granted sufficient restrictions, each type can describe a language equally well; and consequently, that such grammars can be considered "transformational" grammars. The restrictions applied to simple phrase-structure grammar make it sufficient to describe Semitic languages. This grammar has the power to explain the common deep-structure relationships that exist between such forms as the *active* and *passive* voices by showing that they originate from different options of the same symbol. It has the power to explain the universal patterns of a language that transcend the bounds of phrases, and it has a type of *context sensitivity* sufficient for explaining the semantic concord found in natural languages. All of this is provided by a relatively small number of unordered rules without a second system of "transformational" notation. A specific application of this grammar is made to one Semitic language (modern Hebrew) in Part II, and computer tests of the grammar are reported in Parts III and IV which verify these conclusions.

*See Part III, Section 3.3.2.3, for a full description of the method.

REFERENCES

1. Topping, D. M., "Some Implications of Transformational Grammar for Language Teaching," June, 1969, ED 034 982, U.S. Office of Education, ERIC.

2. Harris, Zellig, "Transfer Grammar," *International Journal of American Linguistics*, Vol. 20, No. 4 (Oct. 1954), pp. 259-270.

3. Chomsky, Noam, "Three Models for Description of Languages," *IRE Transactions on Information Theory*, Vol. IT-2, No. 3 (Sept. 1956).

4. Chomsky, Noam, *Syntactic Structures*, Mouton, s'-Gravenhage, Holland (1957).

5. Harris, Zellig, *String Analysis of Sentence Structure*, Mouton and Co., The Hague (1962).

6. Joshi, A., "A Procedure for a Transformational Decomposition of English Sentence," TDAP 42, University of Pennsylvania (1962).

7. Sager, Naomi, "A Procedure for Syntactic Analysis," *Proceedings of the IFIP Congress*, Munich (1962).

8. Sager, Naomi, "A Computer String-Grammar of English," November 1968, ED 030 850, U.S. Office of Education, ERIC.

9. Rhodes, Ida, *Hindsight in Predictive Syntactic Analysis*, National Bureau of Standards Report 7034 (Nov. 30, 1960).

10. Rhodes, Ida, "A New Approach to the Mechanical Syntactic Analysis of Russian," *Mechanical Translation*, Vol. 6 (Nov. 1961).

11. Kuno, S., and Oettinger, A. G. "Syntactic Structure and Ambiguity of English," *Proceedings of the International Federation of Information Processing Congress* (1963).

12. Oettinger, A. G. and Kuno, S., "Multiple-Path Syntactic Analyzer," *Proceedings of the International Federation of Information Processing Congress* (1962).

13. Lindsay, R. K., "Inferential Memory as the Basis of Machines Which Understand Natural Language," *Computers and Thought*, McGraw-Hill (1963), pp. 217-233.

14. Harmon, Gillard H., "Generative Grammar Without Transformation Rules, A Defense of Phrase Structure," *Language*, Vol. 39, No. 4 (Oct.-Dec. 1963).

15. Price, James D., *The Development of a Theoretical Basis for Machine Aids for Translation from Hebrew to English*, Ph. D. dissertation, The Dropsie College, Philadelphia, 1969.

16. Price, James D., "An Algorithm for Generating Hebrew Words," *Computer Studies in the Humanities and Verbal Behavior*, Vol. 1, No. 2, August 1968.

17. Price, James D., "An Algorithm for Analyzing Hebrew Words," *Computer Studies in the Humanities and Verbal Behavior*, Vol. 2, No. 3, 1969.

18. Lenneberg, Eric, "Language, Evolution and Purposive Behavior," *Culture in History: Essays in Honor of Paul Radin*, S. Diamond, Ed., Columbia, 1960.

19. Chomsky, Noam, *Aspects of the Theory of Syntax*, MIT, 1965.

20. Hodge, Carleton T., "Response to W. Gage's Article *Uncommonly Taught Languages*," *ERIC Bulletin No. 17*, Center for Applied Linguistics, Washington, September, 1970.

21. *Language Research in Progress: 9*, Center for Applied Linguistics, Washington, June 1970.

22. Gage, William, "Uncommonly Taught Languages," *ERIC Bulletin No. 17*, Center for Applied Linguistics, September 1970.

23. Kant, Julia G., "Foreign Language Registrations in Institutions of Higher Education, Fall 1968," *Foreign Language Annals*, Volume 3, pp. 247-304, New York: American Council on the Teaching of Foreign Languages, 1969.

24. Lehmann, W. P., and Pendergraft, E. D., *Report No. 23, Seventh Quarterly Progress Report, 1 November, 1964-31, January 1965*, Linguistic Research Center, The University of Texas (May 1966).

25. Lehmann, W. P. and Pendergraft, E. D., *Report No. 27, Eleventh Quarterly Progress Report, 1 November, 1965-31, January 1965*, Linguistic Research Center, The University of Texas (May 1966).

26. *ICRH Newsletter*, Institute for Computer Research in the Humanities, New York University, Vol. II, Nos. 6-7 (February-March 1967).

27. *Language Research in Progress: 9*, Center for Applied Linguistics, Washington (December 1969).

28. *ICRH Newsletter*, Institute for Computer Research in the Humanities, New York University, Vol. II, No. 1 (September 1966).

29. Berman, Lawrence V., in *ICRH Newsletter*, Institute for Computer Research in the Humanities, New York University, Vol. II, No. 1 (September 1966).

30. Ornan, Uzzi, *The Nominal Phrase in Modern Hebrew*, Applied Logic Branch, The Hebrew University of Jerusalem, Technical Report No. 18, U.S. Office of Naval Research, Information Systems Branch, Jerusalem (May 1965).

31. Shapiro, Meyer, and Choueka, Yaacov, "Machine Analysis of Hebrew Morphology: Potentialities and Achievements," *Leshonenu*, No. 27, 1964.

32. Choueka, Yaacov, "Automatic Grammatical Analysis of the Hebrew Verb," *Proceedings of the Second Conference of IPA*, Information Processing Association of Israel, Rehovoth, 1966, pp. 49-67.

33. Choueka, Yaacov, "Automatic Derivation and Inflection in Hebrew Morphology," *ICRH Newsletter*, Institute for Computer Research in the Humanities, New York University for Computer Research in the Humanities, New York University, Vol. IV, No. 11 (July 1969).

34. Choueka, Yaacov, and Yeshuran, S., "Statistical Aspects of Modern Hebrew Prose," *Proceedings of the Fifth National Conference of IPA*, Information Processing Association of Israel, Jerusalem, 1969.

35. *Computational Linguistics*, Asa Kasher and Yaacov Choueka, editors, IPA Publications, General Series, Volume 2, Information Processing Association of Israel (1969).

36. Satterthwait, Arnold C., *Parallel of Sentence-Construction Grammars of Arabic and English*, Ph.D. dissertation, Harvard University (1962).

37. Satterthwait, Arnold C., "Computational Research in Arabic", *Mechanical Translation*, Vol. 7, No. 2 (August 1963).

38. *Hebrew Computational Linguistics*, Bulletin No. 1, Bar-Ilan University, Ramat-Gan, Israel (Sept. 1969), pp. 11-14.

39. Grosu, Alexander, "The Isomorphism of Semantic and Syntactic Categories" *Hebrew Computational Linguistics*, Bulletin No. 1, Bar-Ilan University, Ramat-Gan, Israel (Sept. 1969), pp. 35-50.

40. Hayon, Yehiel, *Relativization in Hebrew: A Transformational Approach*, Ph.D. thesis, University of Texas, Austin, Texas.

41. Hayon, Yehiel, "Relative Clauses with Verbal Predicates," *Hebrew Computational Linguistics, Bulletin No. 3*, Bar-Ilan University, Ramat-Gan, Israel, January 1971, pp. 17-65.

42. Bobrow, Daniel G., "Syntactic Analysis of English by Computers—a Survey," *AFIPS Conference Proceedings*, Vol.24 (1963), pp. 365-387.

43. Hays, G. D., "Basic Principles and Technical Variation in Sentence-Structure Determination," RAND p-1984 (May 1960).

44. Hays, G.D., "Grouping and Dependency Theories," *Proc. of the Natl. Symposium on Machine Translation*, edited by H. P. Edmundson, Prentice-Hall (1961).

45. Ajdukiewicz, "Die Syntaktische Kannexitat," *Studia Philosphica*, Vol. 1 (1935), pp. 1-27.

46. Bar-Hillel, Y., "A Quasi-Arithmetical Notation for Syntactic Description," *Language*, Vol. 29, No. 1 (January-March 1953).

47. Lambeck, J., "On the Calculus of Syntactic Types," *Proceedings of Symposia in Applied Mathematics*, Vol. XII (1961).

48. Bloomfield, Leonard, *Language*, Henry Holt and Co., N.Y. (1933).

49. Friedman, Joyce, *A Computer Model of Transformational Grammar*, American Elsevier, New York, January 1971.

60